



**COMPARATIVE ANALYSIS OF VECTOR QUANTIZATION METHODS
USED IN SPEECH PROCESSING**

HIBA FARAJ ALI FARAJ

DECEMBER 2019

**COMPARATIVE ANALYSIS OF VECTOR QUANTIZATION METHODS
USED IN SPEECH PROCESSING**

**A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED
SCIENCES OF
ÇANKAYA UNIVERSITY**



**BY
HIBA FARAJ ALI FARAJ**

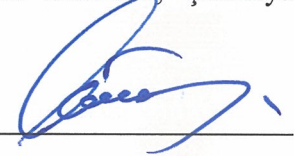
**IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF PHILOSOPHY OF DOCTORATE IN
THE DEPARTMENT OF
ELECTRONIC AND COMMUNICATION ENGINEERING**

DECEMBER 2019

Title of the Thesis: Comparative Analysis of Vector Quantization Methods Used in Speech Processing

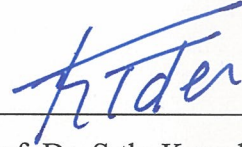
Submitted by **Hiba Faraj Ali FARAJ**

Approval of the Graduate School of Natural and Applied Sciences, Çankaya University.



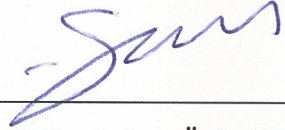
Prof. Dr. Can ÇOĞUN
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Philosophy of Doctorate of Science.



Prof. Dr. Sıtkı Kemal İDER
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Philosophy of Doctorate of Science.

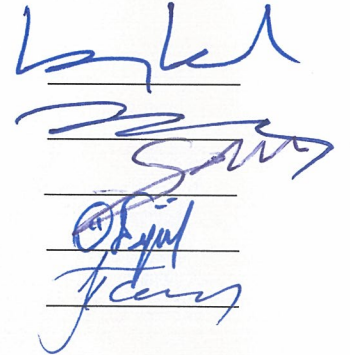


Assist. Prof. Dr. Selma ÖZAYDIN
Supervisor

Examination Date: 06.12.2019

Examining Committee Members


Prof. Dr. Buyurman BAYKAL	(METU)
Assoc. Prof. Dr. Hasan Şakir BİLGE	(Gazi Univ.)
Assist. Prof. Dr. Selma ÖZAYDIN	(Çankaya Univ.)
Assist. Prof. Dr. Özgür ERGÜL	(Atılım Univ.)
Assist. Prof. Dr. Sibel TARIYAN ÖZYER	(Çankaya Univ.)



STATEMENT OF NON-PLAGIARISM PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : HIBA FARAJ

Signature : 

Date : 8.1.2020

ABSTRACT

COMPARATIVE ANALYSIS OF VECTOR QUANTIZATION METHODS USED IN SPEECH PROCESSING

FARAJ, HIBA ALI FARAJ

Ph.D., Department of Electronic and Communication Engineering

Supervisor: Assist. Prof. Dr. Selma ÖZAYDIN

December 2019, 105 pages

Vector quantization techniques play a vital role in compression of speech signals. There are a variety of vector quantization techniques. Each technique has its own advantages and disadvantages and there is no vector quantization technique presenting perfect results in all aspects till now. This thesis deals with enhancing the performance of the existing vector quantization techniques by using new methods. In this thesis hybrid vector quantization techniques which are produced from the existing methods are proposed. The performance of the designed vector quantizers are evaluated in terms of the spectral distortion measured, computational complexity and memory requirements.

In the scope of this thesis, Multistage vector quantization (MSVQ), Split Vector Quantization (SVQ), Residual Vector Quantization (RVQ), Residual Multistage Vector Quantization (R-MSVQ), Residual Split Vector Quantization (R_SVQ) and voiced/unvoiced Residual Multistage Vector Quantization methods (VUV_RMSVQ) are analyzed. Because the VUV_RMSVQ method gave the better test results, further research is directed to find an optimum performance for codebook design with this method. Then, the overall performance of the proposed vector quantization techniques is compared with the existing vector quantization techniques. Whole work is carried out using the standard TIMIT database and both clean and noisy data are tested to

evaluate the performance of the designed codebooks against noise. A linear predictive coding (LPC) based codebook generation algorithm is designed for each vector quantization method. Vector quantization is the process done in between LPC analysis and synthesis. The speech parameters required for vector quantization are the line spectral frequencies (LSF) and are obtained from the LPC coefficients.

At the beginning of the thesis study,-we designed codebooks with MSVQ and SVQ methods and we compared them in terms of spectral distortion. We found that the codebooks with MSVQ method gave better performance. Then, we used the RMSVQ and RSVQ methods to design codebooks. It is seen that the best result was given by RMSVQ. As a result, we continued with RMSVQ and we combined the voiced and unvoiced decision method and RSMVQ technique to achieve better result for spectral distortion. According to the results, it is seen that the best performance is achieved with VUV_RMSVQ method.

ÖZ

KONUŞMA İŞLEMEDE KULLANILAN VEKTÖR NİCEMLEME METOTLARININ KARŞILAŞTIRMALI ANALİZİ

FARAJ, HIBA ALI FARAJ

Ph.D., Çankaya Üniversitesi Elektronik ve Haberleşme Mühendisliği Bölümü

Danışman: Dr. Öğr. Üyesi Selma ÖZAYDIN

Aralık 2019, 105 sayfa

Vektör nicemleme teknikleri konuşma sinyalini sıkıştırmada hayati bir öneme sahiptir. Çok çeşitli vektör nicemleme metotları mevcuttur. Herbir teknik kendine has avantaj ve dezavantajlar içermektedir ve tüm yönleriyle mükemmel sonuçlar veren bir vektör nicemleme metodu henüz yoktur. Bu tez çalışması, mevcut vektör nicemleme tekniklerinin performansını yeni metotlar uygulayarak iyileştirmeyi amaçlamaktadır. Bu tezde mevcut metotlardan hareketle melez vektör nicemleme teknikleri uygulanmıştır. Tasarlanan vektör nicemleyicilerin performansı, spectral distorsiyon, hesapsal karmaşa ve hafıza gereksinimleri bakımından değerlendirilmiştir.

Bu tez çalışması kapsamında Çok aşamalı vektör nicemleme (MSVQ) metodu, Split vektör nicemleme (SVQ) metodu, Artık sinyal vektör nicemleme (RVQ) metodu, ve sesli/sessiz artık sinyal vektör nicemleme metodu (VUV_RMSVQ) analiz edilmiştir. VUV_RMSVQ metodu en iyi test sonuçlarını verdiği için, bu metotla optimum kod tablosu tasarlamada yeni metotlar bulabilmek için araştırma derinleştirilmiştir. Daha sonra, tüm tasarlanan vektör nicemleme metotlarının performansları var olan metotlarla karşılaştırılmıştır. Tüm çalışma standart TIMIT veritabanı kullanılarak ve bu veritabanında temiz ve gürültülü ses verileri kullanılarak yürütülmüştür. Herbir vektör nicemleme metodu için bir Doğrusal öngörülü kodlama (LPC) tabanlı kod tablosu üretim algoritması tasarlanmıştır. Vektör nicemleme LPC analiz ve sentez

arasında gerekleřtirilen bir iřlemdir. Vektör nicemleme için gerekli konuřma parametreleri izgi spectrum frekanslarıdır (LSF) ve bunlar LPC katsayılarından elde edilirler.

Tez alıřmasının bařlangıcında, MSVQ ve SVQ metotları ile kod tabloları tasarladık ve bunları spectral distorsiyon bakımından karřılařtırdık. MSVQ metodu ile tasarlanan kod tablolarının daha iyi sonular verdiđini grdük. Daha sonra, kod tablosu tasarlamak için RMSVQ ve RSVQ metotlarını kullandık. Sonulardan grld ki en iyi sonu RMSVQ metodu tarafından verildi. Sonu olarak, RSMVQ metodu ile devam ettik ve spectral distorsiyon için en iyi performansı bařarabilmek için sesli/sessiz karar metodunu RSMVQ metodu ile birleřtirdik. Test sonularına gre, en iyi performansın VUV_RMSVQ metodu ile bařarıldıđı grld.



ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to Assist. Prof. Dr. Selma ÖZAYDIN for her supervision, special guidance, suggestions, and encouragement through the development of this thesis.

I would like to thank my father, Dr. Faraj Ali for his support and love, and special thank for my husband and family for their support, I would like to thank to my mother, she passed away and did not see my graduation. I wish she was here, but I know she is proud of me, may Allah, have mercy on her soul, and my success is because of her and my father support.

I would like to express my truthful respect for the government of Libya represented by the Ministry of Higher Education and Scientific Research for their financial support to complete my study.

TABLE OF CONTENTS

STATEMENT OF NON-PLAGIARISM.....	iii
ABSTRACT.....	iv
ÖZ.....	vi
ACKNOWLEDGEMENTS.....	viii
TABLE OF CONTENTS.....	ix
LIST OF FIGURES.....	xiii
LIST OF TABLES.....	xvii
LIST OF ABBREVIATIONS.....	xviii

CHAPTERS:

1. INTRODUCTION	1
1.1. INTRODUCTION.....	1
1.2. Speech production modeling.....	2
1.2.1. Mechanism of speech production.....	3
1.3. A General Structure of Analysis Speech Coding/Decoding Synthesis..	5
1.4. Speech Coding.....	6
1.5. Thesis Contribution.....	7

1.6.	Thesis organized.....	8
2.	LITERATURE REVIEW	10
2.1.	Literature Review.....	10
2.2	Turkish thesis literature search about vector quantization.....	13
3.	LINEAR PREDICTIVE IN A SPEECH CODING SYSTEM	15
3.1.	Windowing.....	15
3.1.1.	Window Length.....	18
3.2.	Linear Predictive Coding (LPC).....	19
3.2.1.	The LPC Order Definition.....	20
3.2.2.	Calculation of the LPC Gain.....	21
3.2.3.	Calculation of the LPC Coefficients and Quantization.....	21
3.2.4.	Conversion of LPC to LSF Coefficients and Quantization...	22
3.2.4.1.	Interpolation of the LPC Coefficients.....	22
3.2.4.2.	All Pole-Zero Modelling of Speech.....	23
3.2.4.3.	Correlation of Linear Predictive Coding.....	24
3.2.4.4.	LPC Analysis Filter.....	25
3.3.	Line Spectral Frequency (LSF).....	26
3.4.	Distortion Measures of Quantized LSF Coefficients.....	28
3.4.1.	Spectral Distortion.....	28
4.	DESIGN OF CODEBOOKS IN VECTOR QUANTIZATION	30
4.1.	Quantization Theory.....	30
4.1.1.	Scalar Quantization.....	31

4.1.2.	Scalar Quantization of LPC parameters	32
4.1.3.	Scalar Quantization using LPC representation	32
4.2.	Vectorial Quantization	34
4.2.1.	Design of codebooks	37
4.2.2.	Optimality Criteria	38
4.2.3.	Nearest Neighbor Condition	38
4.3.	Codebook Generation Algorithms	39
4.3.1.	Full Search Codebook	39
4.3.2.	Binary Search Codebook	40
4.3.3.	LBG Algorithm	42
4.3.3.1.	LBG Algorithm Design	42
4.3.3.2.	Flow chart for LBG Algorithm	45
4.4.	Training, Testing and Codebook Robustness	46
5.	VECTOR QUANTIZATION METHODS	47
5.1.	Introduction	47
5.2.	Multistage Vector Quantization (MVQ)	48
5.3.	Split Vector Quantization (SVQ)	52
5.4.	Residual Vector Quantization	54
6.	CODEBOOK DESIGN WITH MSVQ AND SVQ METHODS	56
6.1.	Introduction	56
6.2.	Split Vector quantization codebook design	57
6.2.1.	Performance Analysis for Split Vector quantization	65
6.3.	Multistage Vector Quantization Codebook Design	62

6.3.1.	Performance Analysis for Multistage Vector Quantization.....	63
6.4.	Distortion, Complexity and Memory measurement for the codebooks.	64
6.4.1.	Complexity.....	65
6.4.2.	Memory measurement.....	65
6.5.	Performance Comparison of MSVQ and SVQ codebook Results.....	76
7.	RESIDUAL LSF QUANTIZATION METHODS	68
7.1.	Introduction.....	68
7.2.	Design of Residual MSVQ and Residual SVQ Codebooks.....	70
7.3.	Residual Multistage Vector Quantization.....	71
7.4.	Residual Split Vector Quantization SVQ.....	74
7.5.	Performance Evaluation.....	77
8.	DESIGN OF RMSVQ CODEBOOK WITH VOICED/UNVOICED EVALUATION	83
8.1	Introduction.....	83
8.2	Voiced/Unvoiced Evaluation of Speech Signal.....	84
8.2.1	Zero crossing rate.....	84
8.2.2	Short-time energy.....	85
8.3	Speech Voiced/Unvoiced Classification.....	86
8.3.1	Algorithm design for the proposed method.....	86
8.3.2	Voiced/Unvoiced multistage vector quantization codebook design.....	88
8.3.3	Codebook search and quantization.....	89
8.4	Residual multistage vector quantization for Voiced/Unvoiced Speech using different size of codebooks.....	90

8.5	Residual multistage vector quantization for Voiced/Unvoiced Speech using fixed size of codebooks(2-4) stages.....	91
8.6	Residual multistage vector quantization for Voiced/Unvoiced Speech using fixed size of codebooks(3-4) stages.....	92
8.7	Performance Analysis.....	92
9.	CONCLUSION	95
9.1	Conclusion.....	95
9.1.1	First method evaluation.....	96
9.1.2	Second method evaluation.....	96
9.1.3	Third method evaluation.....	97
	REFERENCES.....	R1

LIST OF FIGURES

FIGURES

Figure 1	Human speech production model.....	3
Figure 2	Speech coding/decoding.....	5
Figure 3	Time plots of various window frequency	17
Figure 4	Frequency response of various window function.....	18
Figure 5	Block diagram of LSF coefficients calculation and quantization in speech analysis part.....	19
Figure 6	Poles and Zeros.....	24
Figure 7	A non-uniform quantizer.....	30
Figure 8	Uniform quantizer.....	31
Figure 9	1-dimensional Vector Quantizer.....	35
Figure 10	2-dimensional Vector Quantizer.....	34
Figure 11	Binary splitting into eight cells.....	39
Figure 12	Flow diagram of the LBG algorithm (Adapted from Rabiner and Juang, 1993).....	42
Figure 13	Codebook generation for different stages of MSVQ.....	47
Figure 14	Block diagram of three part split vector quantizer.....	50
Figure 15	Flow chart for SVQ.....	56
Figure 16	Flow chart for MSVQ.....	59
Figure 17	3-Stage MSVQ and 3-split SVQ	61
Figure 18	Complexity in MSVQ and SVQ	63

Figure 19	Memory Requirement in MSVQ and SVQ.....	65
Figure 20	Flow chart for RMSVQ.....	66
Figure 21	Flow chart for RSVQ.....	66
Figure 22	SD for 3-stage RMSVQ and RSVQ	71
Figure 23	SD for 4-stage RMSVQ and RSVQ	73
Figure 24	SD for 3- stages RSVQ and SVQ.....	75
Figure 25	SD for 3- stages RSVQ and SVQ.....	77
Figure 26	SD for 3- stage RMSVQ and MSVQ.....	78
Figure 27	SD for 4- stage RMSVQ and MSVQ.....	78
Figure 28	Memory requirement comparison for all stages in (SVQ and MSVQ).....	81
Figure 29	Complexity requirement comparison for all stages in (SVQ and MSVQ).....	79
Figure 30	Definition of zero-crossing rate.....	79
Figure 31	Definition of short time energy.....	81
Figure 32	Flow chart for VUV_RMSVQ.....	82
Figure 33	VUV_RMSVQ Block diagram.....	86
Figure 34	Block diagram for VUV_RMSVQ using two and four stages...	87
Figure 35	Flow chart for VUV_RMSVQ 2 and 4 stage.....	89
Figure 36	SD for VUV_RMSVQ for 2 and 4 stage	91
Figure 37	SD for VUV_RMSVQ for 2 and 3 stage.....	91
Figure 38	Comparative between MSVQ, RMSVQ and VUV_RMSVQ using noisy speech.....	92
Figure 39	Complexity for VUV_RMSVQ.....	93
Figure 40	Memory for VUV_RMSVQ.....	93

Figure 41	Comparative between increasing bit number for voiced and unvoiced for VUV_RMSVQ.....	94
Figure 42	Comparative between MSVQ,RMSVQ and VUV_RMSVQ for clean speech.....	95



LIST OF TABLES

TABLES

Table 1	SD for 2-split (SVQ) using clean speech.....	56
Table 2	SD for 3-split (SVQ) using clean speech	56
Table 3	SD for 3-split (SVQ) using noisy speech.....	57
Table 4	SD for 3 and 4 stages (MSVQ) using clean speech.....	60
Table 5	SD for 3 and 4 stages (MSVQ) using noisy speech.....	61
Table 6	SD for RMSVQ using three and four stages using clean speech.....	72
Table 7	SD for RSVQ for three split using clean speech.....	74
Table 8	SD for RSVQ for four split using clean speech.....	75
Table 9	SD for RSVQ and RMSVQ for three split using noisy speech...	75
Table 10	SD for VUV_RMSVQ	86
Table 11	SD for VUV_RMSVQ using two and four stages.....	89
Table 12	SD for VUV_RMSVQ using two and four stages.....	89
Table 13	SD for VUV_RMSVQ (increasing and decreasing number of bits for voiced and unvoiced).....	93
Table 14	Results for VUV_RMSVQ (increasing and decreasing number of bits for voiced and unvoiced).....	94

LIST OF ABBREVIATIONS

+	Addition
−	Subtraction
*	Multiplication
/	Division
<	Less than
>	Greater than
Σ	Summation
Π	Product
\int	Integral
$\sqrt{\quad}$	Square root
$\phi(i, j)$	Covariance coefficient of (i, j)
α_j	Prediction coefficients
β	Pitch gain
τ	Lag or Delay
$[\]^T$	Transpose of a matrix
ε	Threshold level
B	Number of bits allocated to a quantizer
$E(\tau)$	Error with lag
$e^2(n)$	Mean square error
f_1	Lower frequency

f_2	Upper frequency
G	Gain of filter
Kbps	Kilobit per second
$H(z)$	Synthesis filter
L	Length of the codebook
LPC	Linear Predictive Coefficients
LPCC	Linear Prediction Cepstral Coefficients
LSF	Line Spectral Frequencies
LBG	Linde Buzo Gray algorithm
m	Number of switches
MSVQ	Multistage Vector Quantizer
SVQ	Split Vector Quantizer
RVQ	Residual Vector Quantizer
R_MSVMQ	Residual Multistage Vector Quantizer
RSVMQ	Residual Split Vector Quantizer
b	Number of bits
C	Codebook
C_j	i^{th} codeword in a codebook
D	Difference between two vectors
D_W	Weighted Distance measure
DFT	Discrete Fourier Transform
MSE	Mean Square Error
MFCCs	Mel Frequency Cepstral Coefficients
N	Number of vectors or code words in a codebook
n	Dimension of a vector
p	Order of filter
P	Number of stages

PS	Power spectrum
$R(j)$	j^{th} auto correlation coefficient
R	Region
$\hat{s}(n)$	Synthetic Speech samples
S	Vector of a speech signal
$E[s]$	Expectation of the random variable s
S	Static weights
$S(n)$	Output speech signal
S_p	Number of splits
\hat{S}	Quantized vector of a speech signal
SD_i	Spectral distortion of i^{th} frame
T	Sampling period
TV	Training sequence
TIMIT	Texas Instruments and Massachusetts
T_0	Pitch period
UV	Unvoiced segment
U	Volume velocity
V	Voiced segment
W	Dynamic weights
ZCR	Zero Crossings Rate

CHAPTER 1

1.1 Introduction

Speech signals are unique for various causes, of which the most important one is that they are in motion, thus making them challenging for simulation and analysis. Next, accuracy, consistency and similar other attributes are key in such analyses. A third factor is the discrete values necessary for defining a second of such signals being equivalent of a minimum of 8000 bits. Given that the frequency impacts processing costs, these indicators become compacted prior to their transmission. In this respect, speech coding obtains a rather condensed representation of these signals in order to effectively send them across frequency-specific wired or wireless media as well as proper recording.

These days, coders have turned into the inseparable part of telecom and media industries given that the type of frequencies applied decide the costs associated. The aim is to simulate signals using the least amount of bits and without loss of perceptual attributes. Coding allows telecom firms to include more calls within one fiber link or cable; apart from this, it is crucial for mobiles and cell phones considering the data limits for each user - the less this value for each call, the more the other utilities [1-2]. Other benefits include voice-over IP, videoconferences, and multimedia by decreasing the bandwidth needs on the Web [3]. Furthermore, many of such applications call for minimum delay as; otherwise, the flow of conversation can experience late transmission and delay [4]. Coders transform digitized signals to coded figures and sends them as frames. The receiver then decodes these frames for further synthesis so as to recreate the signals. These coders vary based on in bit-rate, complexity, delay and comprehensibility of the outcome [5]. In particular, two major categories are at

work: narrowband speech coding and wideband speech coding. The former codes the signals in between 300 to 3400 Hz with 8 kHz as the sampling rate, whereas the latter does so under 50 to 7000 Hz with 14–16 KHz as the sampling rate. Narrowband coding has wider use compared to wideband due to the very quality of phone lines at about 300 to 3400 Hz [6]. Nowadays, wideband coding has been gaining more popularity for other purposes as video conferences.

The present study is in accordance to a conventional TIMIT database, following the stages below:

- Speech analysis: to include framing, windowing, frame overlapping, determining linear predictive coefficient, frame pitch assessment, and linear predictive coefficients (LPC) transformation to line spectral frequencies factors (LSF).
- Vector quantization: to include codebook design with the help of Linde, Buzo, and Gray algorithm, different vector quantizers design, and using vector quantization (VQ) on line spectral frequencies (LSF).
- Speech Synthesis: to include modifying LSFs to LPCs and, later, synthesis with the help of pitch, gain, and quantized LPC parameters.
- Calculating spectral distortion, outliers, and unsteady frames.

The motivation behind this work is to develop an efficient vector quantizer having low bit-rate, complexity and memory requirements when compared to the existing techniques and to best suit it for applications involving speech coding.

1.2 Speech production and modeling

Speech signals possess a series of sounds, which, along with the transitions, serve as information-carrying signs. Their sequence is based on linguistic patterns hence the scientific term for studying the discipline. Another field dealing with sound forms and their development is the phonetics.

1.2.1 The Mechanism of Speech Production

Speech waves are generated based on certain bodily organs and their motions to make up human speech, whose formation mechanism is illustrated in Figure 1 [16]. No matter what language one speaks, every individual applies more or less the same

organs to generate speech. These productions follow the principles of physics and, in brief, can be explained as air forced from the lungs, passing the vocal tract and out via the oral cavity to form a sound or speech. Here, the lungs can be regarded as where the sound begins, with the vocal tract as the filter to form different sounds to constitute speech as a whole.

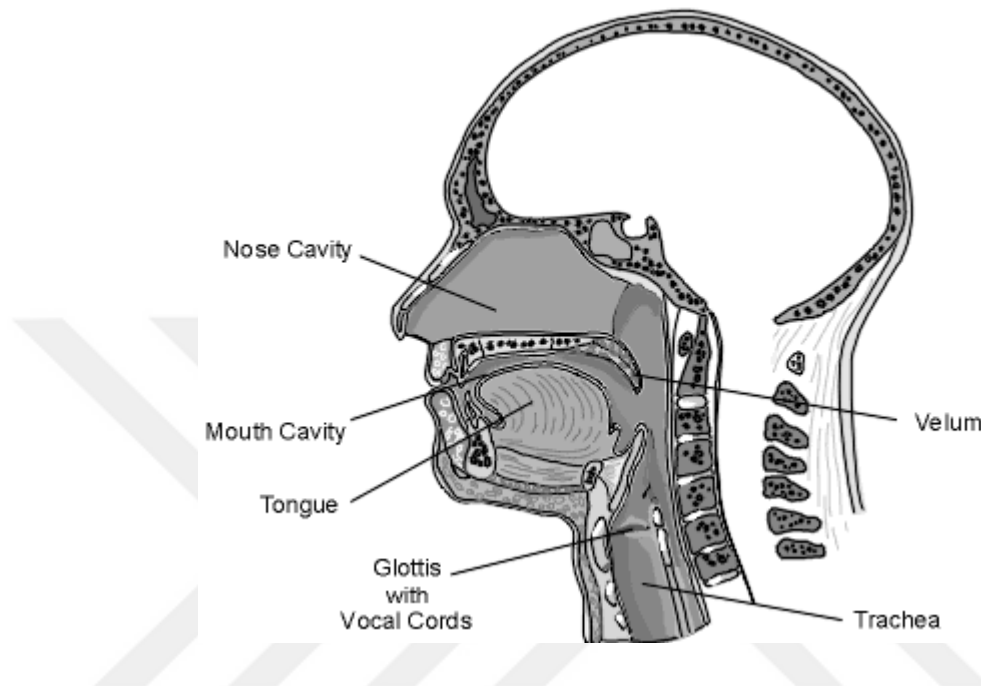


Figure 1 .Human speech production model

To clearly comprehend the role of the vocal tract in changing air into sound, certain concepts come into view. To begin with, phonemes are a defined series of sounds, mainly grouped into two voiced, unvoiced, and detected by coders once signal analysis and synthesis is carried out.

Voiced sounds are of higher energy values, distinguishable form, and profound frequency generated once air periodically vibrates the vocal cords to create a series of signals otherwise known as glottal pulses. The vibration rate as regards the vocal cords determines the pitch. These signals eventually travel along the vocal tract, where certain frequencies echo and resonate. While being generated by women and children, these signals possess higher pitch values compared to men owing to a speedy vibration of the vocal cords. Henceforth, to have satisfactory results for accuracy in original

input representation, all information related to the pitch period has to appear in the analysis and synthesis. Voiceless sounds occur more frequently due to more resonant frequencies and reduced energy compared to voiced ones; they are also formed once unstable flows of air travel along the vocal tract, rendering the vocal cords free of vibration and to maintain an open status until the sound is generated. Pitch is a key feature in unvoiced signals due to the absence of vocal cord vibration and signals [17]. Classifying sounds as voiced or unvoiced is another key issue once analysis and synthesis is conducted; indeed, vocal cords vibration or limited vibration constitutes a major aspect of generating various sounds. A separate factor affecting speech the anatomy of the vocal tract since various sounds or resonant frequencies can be generated as a result. The tract comprises, in detail, the throat, tongue, nose, mouth and lips described as the route by which speech is formed and various frequencies can be generated. During speaking, this tract is on a slow but steady transformation of shape to create various sounds, which eventually come together and form words.

Lastly, the flow of air moving from the lungs can influence speech production given its role as the source of all signals. As stated before, the vocal tract is a filter to generate speech by taking air from the lungs; the more air, simply the louder the sound [18].

1.3 A general structure of analysis speech coding /decoding synthesis

During the coding process, the primary input signals, being analog, are digitized with the help of a medium filter, sampler and analog-to-digital (A/D) transforming circuits. This is an anti-aliasing filter, basically with low-pass, in use prior to a sampler so as to eliminate those frequencies beyond the Nyquist standard. Filtering prevents aliasing and should the sampling frequency fall lower than two times the size of the subject signal bandwidth, then aliasing is inevitable. To solve the problem, the sampling frequency can be dialed up to values more than 2.5 times the analog sample bandwidth and in accordance to the assertion by Nyquist theory. About 8 KHz can be set as the conventional sampling frequency for telephone signals as they vary from 300 to 3400 Hz [5]. Next, the sampler changes the analog signals to discrete patterns of input to an A/D modifier to produce digitized patterns. The available coding mechanisms are crafted for telecom support services as they restrict the frequency contents between 300 and 3400 Hz. To change analog signals to digital ones, keep high quality and

create signals identical to the original input, they have to be tested with over 8 bits per sample. Figure 2 represents a block diagram for speech coding.

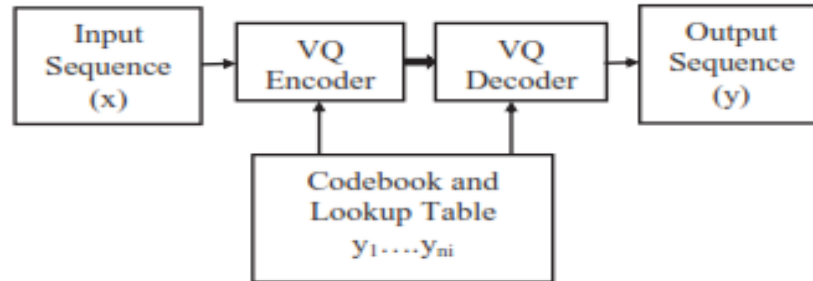


Figure 2. Speech analysis steps

In the present study, the factors related to digital signals comprise the sampling frequency of 8 kHz and 8 bits per sample. Accordingly, the input signal has a bite rate of 64 Kbps. Encoders serve to decrease the input bit-rate under 64 Kbps, beyond which all bit-rates are taken as compression, and the encoder output has a bit-rate under 64 Kbps. Related algorithms for our purpose of decreasing this bit-rate are of LPC nature, based upon which bit-rate is downsized to 1 Kbps that is to say, a 64-fold reduction compared to the input. Essentially, the source encoder output becomes the input to the channel encoder to prevent errors in the bit flow sent across the channel, where noise and other impediments may hamper the transferred data accuracy.

On the receiver's side, the channel decoder obtains the encoded data from the error-protected data to supply to the source decoder to gain the actual signal, which is then supplied in turn to a D/A transformer so as to change signals from digital to analog. Lastly, this analog content is supplied to an aliasing-proofer filter to prevent such incidents when constant signals are being recreated, thus calling for a solid stop-band prevention mechanism to achieve maximum aliasing-free content [5,7-9].

1.4 Speech coding methods

Coding essentially condenses the signals through the reduction of bits for each sample, thereby making the decoded speech differentiable in auditory terms. In detail, coding is carried out to fulfil these goals [11]:

- Reduction in bit-rate or equivalently the bandwidth.
- Decreasing the memory requirements, which decreases in a proportionate manner with respect to the bit-rate.
- Decreasing the transmission power required, since compressed speech signal has less number of bits per second to transmit.
- Immunity to noise, some of the saved bits per sample can be used as protective error control bits to the speech parameters.

The approaches to coding are largely grouped as lossless and lossy; in the former, the recreated signal by the decoder is identical in shape to the input signal. However, in the latter case, the recreated signal is perceptually the same as the source signal. Also, in the lossy approach, despite the recreated waveform's difference from the original one, most coding methods circle around the lossy approach and eliminate non-essential information in terms of perceptual quality. Today's coding techniques comprise waveform coding as well as parametric coding.

1.5 Thesis Contribution

The original contribution of this study is to search of new techniques for design of the vector quantization codebooks to satisfy a reasonable performance in different conditions such as clean and noisy speech input. In this scope, firstly we applied the MSVQ & SVQ techniques to design codebooks. With these techniques, we designed and tested different sizes of codebooks. Then, we evaluated their performances on large-scale input data for both clean and noisy inputs. Secondly, we applied the residual technique to the MSVQ & SVQ codebooks. We tested the effect of residual technique on these methods by using the same input data and by designing the similar sizes of codebooks with the previous method. We compared the results of MSVQ & R_MSVMQ as well as SVQ & R_SVMQ in terms of spectral distortion (SD) and memory requirements. From the results, it has been found that R_MSVMQ has given better performance in all conditions and for different sizes of codebooks. From these results, we decided to go forward with R_MSVMQ and investigate a robust method increasing the performance of the R_MSVMQ codebooks. After a literature review, we have

decided to separate the input speech signal into voiced and unvoiced parts to increase the performance of codebooks. We have designed three methods as followings;

- first method was separating the speech by using two techniques which are zero crossing rate and short time energy after separating the speech we have used one codebooks for both voiced and unvoiced speech then in test part calculating the SD was done for each one, for voiced we have used fixed value for alpha which is equal to 0.359 and for the unvoiced part we have used matrix A, and calculating the SD for both voiced and unvoiced,

Second technique was using two separated codebooks in the training and testing stage but we hold the both codebook sizes (for voiced and unvoiced) same. The result gave us better SD results when we compared the first method.

In the third technique, we used changing codebook sizes for voiced and unvoiced conditions, as a result decreasing the number of bits in the codebook. It gave a high performance for SD. All programs were tested using clean and noisy TIMIT files and SD and computation complexity and memory requirement was calculated for all programs.

1.6 Thesis Organized

This thesis is outlined as follows. In Chapter 2 Literature Review is introduced.

In Chapter 3 we explained how the speech signal is analyzed stage by stage and dividing the speech into samples and calculating the linear predictive coding (LPC) and the transforming them to linear spectral frequency (LSF) for more stability and finally calculating the spectral distortion for LSF coefficients. In chapter 4 we explained the design of the codebooks in vector quantization (VQ) and explanation why we used VQ and not scalar quantization (SQ) codebook design has been explained that many algorithm has been proposed as seen in the literature review and in our design we have used LBG algorithm in our design. Chapter 5 is a brief description of the important methods that has been used in this thesis and in VQ. Chapter 6 explains who codebooks were designed for both MSVQ and SVQ using large data and using different test data for different type of data which are noisy and clean speech data. Chapter 7 residual vector quantization was applied for both methods split and multistage vector quantization. Chapter 8 in this chapter we applied voiced/unvoiced

method on residual multistage vector quantization and calculated SD and outliers.
Chapter9 presented conclusion and future work.



CHAPTER 2

LITERATURE REVIEW

2.1 Literature Review

This chapter deals with the study of literature in doing this work. The literature studied is explained in detail by highlighting the points concluded in each paper. The references included in this thesis report are all explained in detail. In this section, the background literature is reviewed for the thesis by means of pointing to details related to conclusions derived by other experts and corresponding references.

Speech coding entails the formation of condensed signals through decreasing the bits applied for samples and maintaining perceptual quality, as per reported by A.S. Spanias in 1994 [1], K.Sayood in Introduction to Data Compression 1996 [2], Saeed V. Vaseghi in Multimedia Signal Processing: Theory and Applications in Speech, Music and Communications in 1996[3], John C. Bellamy in Digital Telephony in 1999 [4] and Wai C.Chu in Speech coding algorithms [5]. The background of different coding approaches and the features related to signal and performance criteria in coding are thoroughly dealt with by W.B.Kleijn and K.K.Paliwal in 1995 [6].

Makhoul in 1975 [15] elaborated on linear predictive coding as a method for encoding analog signals with values for available samples to establish a mixture of previous samples, equal to simulating signal spectrums as all pn an all-zero fashion in the bandwidth. Gunnar Fant in Acoustic Theory of Speech Production 1970 [16], Rabiner and Schafer in Digital Processing of Speech Signals 1978 [17], Paget in Human Speech 1999 [18] and Al-Akaidi in Fractal Speech Processing 2004 [19] all describe human speech formation, use of LPC, its filter and its connection to the vocal tract, and pitch details.

Speech analysis entails signal detailing of factors including pitch, gain, and voiced & unvoiced decisions within frames – an effective method according to Pei Hongwen and Shen Fengji [20], R. McAbulay and T. Quatieri [21], Chin-Hui Lee [22], F.F. Tzeng [23], S. Zahorian and P. Gordy [24] and B.Yegnanarayana and P. Satyanarayana Murthy [25]. Obtaining LPCs, their application in analysis and synthesis in the form of filter coefficients, and transforming those LSFs have been addressed with satisfactory information in [26-28]. Voiced and Unvoiced frame decisions, factors and parameters concerning those decisions, mode of determination and arriving at decisions appear with graphic representations as well as lock diagrams in [29-32].

In [33-40], pitch determination in case of both voiced and unvoiced signals appears by means of various techniques, elaborating on the pitch significance in synthesizing signals. Such a process, accordingly, calls for recreating a signal from the parameters collected, among them gain, LPCs, voiced & unvoiced decisions for frames, and pitch. These are key factors for synthesis, which procedure is properly introduced in [41-44].

LP coders' outstanding feature is the quantizer for advanced bit-rate vectors to be turned into lower ones. C.E. Shannon 1959 [45] asserts that quantizing a series of vectors proves to be more beneficial compared to doing so on single values since, once a vector is extensive, the rate distortion function approaches the Shannon limit. The work in 1998 by Gray and Neuhoff [46] offers a review of quantization since the beginnings along with details of different techniques in this field. In a separate work, quantizers able to generate intricate outcomes – also in case of smaller sizes of vectors – come to the fore and discussed by Makhoul, Roucos and Gish in 1985 [47], T.F.Quatieri in Discrete-Time speech signal processing 2004 [48], So.Stephen and K.K.Paliwal in 2007 [49].

J. Tribolet, P. Noll, B. McDermott and R. Crochiere in 1978 [7] and B.S.Atal in 1982[8] revisited adaptive predictive coding focused on obtaining advanced quality with reduced bit-rates, and elaborated further on the impacts related to lower bit-rates on the regenerated signal. N.Kitawaki, H.Nagabuchi and K.Itoh in 1998 [9] introduce the criteria decisive for reduced bit-rate coding and different distance qualities.

Toward the beginning of 1975, quantization attributes were defined by R. Viswanathan and J. Makhoul [59]. Also, the benefits of LSFs related to vector

quantization across LPCs are dealt with alongside their degree of durability in the work by J. Zhou, Y.

Shoham and A. Akansu [60], A.D. Subramaniam and B.D. Rao [61], F.K. Soong and B.H. Juang [62] and W.R. Gardner and B.D. Rao [63] address the VQs and other numerous benefits as opposed to scalar quantizers concerning gap-fills, form and memory attributes. Such benefits reemerge in the work by So. Stephen and K.K.Paliwal in 2007 [49]. Accordingly, many VQ mechanisms are at hand – among them, Unconstrained Vector Quantization applied to full-scale vectors. Not quite popular, the approach ranks high on computational and memory constraints. [64-66].

These restrictions in practice are addressed largely by means of product code VQ methods for codebook search and storage by means of separately conducting single VQ tasks, J.Sabin and R.M.Gray in 1984 [67] address the premises concerning the issue of computation and memory by illustrating standards of quality for coders, emphasizing the benefit of product code VQ mechanisms. In this way, the initial mechanism is the Split Vector Quantizer (SVQ), where vectors with larger sizes are simply shrunk via vector division to tinier scales. SVQ initially appeared in the work by K.K.Paliwal & B.S.Atal in 1993 [68] as a means of tackling complexity. C.S.Xydeas and C.Papanastasiou in 1995 [69], later suggested furthering SVQ intricately and upon gaining an LSF matrix to reach single sub-matrices. In 2004 F.Norden and T.Eriksson [70] discuss the disadvantages of reduced sizes gained by such splitting, further recommending limited quantization to compensate for them. In detail, these disadvantages are alleviated using a Multistage Vector Quantization (MVQ) method to tackle complexity and memory issues.

In 1982 B.H.Juang and A.H.Gray Jr [71] came up with the MVQ approach to further add to unlimited VQ formats, while elaborating on increased bit-rate operation without any addition to noise and the mechanism to decrease computational constraints and memory conditions. W.Y.Chan, S.Gupta and A.Gersho in 1992 [72] came up with an approach related to codebook preparation in every step so as to improve the general performance; the results, though were not quite satisfactory and the encoding and storage matters somehow stayed unimproved. W.P.LeBlanc, B.Bhattacharya, S.A.Mahmoud and V.Cuperman in 1993 [73] came up with a tree-searched MVQ design and an updated and combined codebook plan, which offer spectral distortions

lower than 1 dB and increased the rate of convergence. The work also addressed certain performance criteria for VQ application. J.Pan in 1996 [74] suggested a double-step pyramidal lattice VQ method via applying the tree structure in the initial step to handle the complexity. In 2004 V.Krishnan, D.V.Anderson and K.K.Truong [75] illustrate a channel-optimized Multistage Vector Quantization (COMSVQ) codec, where all codebooks can be arranged in combination. In this way, the regenerated signal quality is developed while decreasing the mean of the spectral distortion. Such noise, computational issues, and memory needs in MVQ are all downsized by means of another method - Split-Multistage Vector Quantizer or S-MSVQ – as combination of SVQ and MVQ techniques, as per the work by So.Stephen and K.K.Paliwal in 2007 [103]. The K-mean algorithm was proposed by MacQueen 1967.it is well known iterative procedure for solving the clustering problems. It is an algorithm to group data into K number of groups by minimizing the sum of the square of distance between data and the corresponding cluster centroid each cluster is represented by the mean of the cluster. [115] Applied techniques to speech-like waveform and used a codebook consisting of vectors in the form of reflection coefficients. They tested the codebook on several speakers and a good result was reported. [117] presented an effective vector quantizer for very low bit rate speech coding based on vector quantization for LSF parameters. As a compromise solution to reduce the storage and searching complexity of the codebook, various structured VQ algorithm such SVQ, DSVQ and mean-removed DSVQ are proposed. The precision was considered for cookbook storage. Spectral distortion (SD) is the most preferred objective measure to evaluate the performance of quantizer. Average spectral distortion (ASD) is proposed to measure the overall quality of spectral distortion.

Liefu Ai , Junqing Ya, Zebin Wu,Yunfeng He and Tao Guan an optimized residual vector quantization is presented for improving the quality of vector quantization and approximate nearest neighbor search.(2015). In (2016) Mobin Jamali, Vahid Ghafarinia and Mohamed Ali Montazeri have proposed a new fast search algorithm to enhance the computational cost and the execution time of the conventional VQ methods.

[117] Hanwer wa, Qiwen Wang and Markus Flierl tree structure vector quantizer was proposed that hierarchically cluster the data into Ksphere-shape quantization cells(2017)

Yi Guo, jia Ye, Lianshan Yan, Wei Pan, Xihva Zou and Hui Yang two-dimensional vector quantization scheme utilizing vector linear prediction for digitalized radio over-fiber(2018).

2.2 Turkish thesis literature search about vector quantization

These are some thesis that has been done in vector quantization in Master and PhD in Turkish

In speech processing

Name	Year	Subject of thesis	University	Type of degree
METIN UZUNCARSILI	2005	Investigation of vector quantization based speaker identification algorithm	Ankara University	Master's Thesis
OSMAN DEMIRHAN	1998	Vector quantization using artificial neural networks	Kocaeli University	Master's Thesis
ONUR ENGIN.T	1998	Sub-band decomposition vector quantization architectures for coding speech and audio signals	Boğaziçi	Master's Thesis
IBRAHIM AVCI	1993	Speech compression based on vector quantization	Boğaziçi	Master's Thesis
MEHMET SEDAT.U	1991	A study on the application of vector quantization to sub-band coding of speech signals	Orta doğu teknik university	Master's Thesis

In image processing

Name	Year	Subject of thesis	University	Type of degree
Şebnem KOLTAN.Y	2014	Control charts pattern recognition based on linear vector quantization neural networks	Inönü unversity	Ph.D. Thesis
ICLAL GOR	2013	A design and implementation of geometrical learning algorithm for vector quantization	Adnan Menderes University	Master's Thesis
ERKAN KOCAKAYA	2005	Video compression with vector quantization	Kocaeli University	Master's Thesis
ÖZGÜR ORACAY	2003	3-D object mesh geometry compression with vector quantization	Boğaziçi	Master's Thesis
ILKER KILIÇ	2003	A video compression algorithm using zerotree wavelet and hierarchical finite state vector quantization	Dokuz Eylül University	Ph.D. Thesis
MEHMET YAKUT	2002	Image compression using vector quantization for nonsquare blocks	Kocaeli University	Ph.D. Thesis

CHAPTER 3

LINEAR PREDICTIVE IN A SPEECH CODING SYSTEM

Speech signal analysis is used to characterize the spectral information of an input speech signal. Speech signal analysis [12] techniques are employed in a variety of systems, including voice recognition and digital speech compression. Popular method of analyzing speech signals uses linear predictive coding (LPC). In linear predictive coding, each sample of a digital input speech signal is represented as a combination of an innovation sample and a weighted series of past speech samples. The series coefficients, or weights, are referred to as LPC coefficients. Real-time LPC analysis of speech signals is a computationally burdensome process.

Many voice recognition systems currently use LPC speech analysis techniques to generate useful spectral information about an input speech signal. In voice recognition, LPC techniques are employed to create observation vectors, which are used by voice recognizers. These observation vectors are compared or matched to stored model vectors in order to recognize the input speech. Voice recognition systems have been utilized in various industries, including telephony and consumer electronics. For example, mobile telephones may employ voice recognition to allow "hands free" dialing, or voice dialing. In the analysis of the speech signal, first the windowing technique is implemented.

3.1 Windowing

The window, $w(n)$, determines the portion of the speech signal that is to be processed by zeroing out the signal outside the region of interest. The ideal window frequency response has a very narrow main lobe which increases their resolution and no side lobes (or frequency leakage). Since such a window is not possible in practice, a compromise

is usually selected for each specific application. There are many possible windows (e.g. Rectangular, Bartlett, Hamming, Hanning, Blackman, Kaiser, etc.), some of which are defined as follows:

In order to smooth the estimated power spectrum and to avoid abrupt transitions in frequency response between adjacent frames, one may choose to use some kind of windowing function in the linear analysis. Spectral smoothing techniques are used to avoid distinct peaks in the spectrum, which will result in poles near the unit circle. The effect of multiplying the input with a finite-length window is equal to convolving the power spectrum with the frequency response of the window. This causes the side-lobes in the frequency response of the window to have an averaging effect on the signal power spectrum. The use of 160-sample frames in the linear analysis would be equal to windowing the input with a 160-point rectangular window. In the analysis, we use a 160-point Hamming window, which has better frequency properties than the rectangular window. The effect of this is to produce a weighted average of the input, where the 160 samples in the center of the Hanning window correspond to the frame being processed, i.e. the last sub-frame of the preceding frame and the first one of the next frame are also included in the analysis. This alleviates the effect of abrupt transitions in the frequency properties of adjacent frames

There are many possible windows (e.g. Rectangular, Hamming, Hanning, Blackman, etc.), some of which are defined as below:

Rectangular:

$$w(n) = \begin{cases} 1: & 0 \leq n \leq N - 1 \\ 0: & \textit{otherwise} \end{cases} \quad (3.1)$$

Blackman:

$$w(n) = \begin{cases} 0.42 - 0.5 \cos\left(2\pi \frac{n}{N-1}\right) + 0.8 \cos\left(2\pi \frac{2n}{N-1}\right) : & 0 \leq n \leq N - 1 \\ 0 & \textit{otherwise} \end{cases} \quad (3.2)$$

Hanning:

$$w(n) = \begin{cases} 0.5 - 0.5 \cos\left(2\pi \frac{n}{N-1}\right) & : 0 \leq n \leq N-1 \\ 0 & : \text{otherwise} \end{cases} \quad (3.3)$$

Hamming:

$$w(n) = \begin{cases} 0.54 - 0.46 \cos\left(2\pi \frac{n}{N-1}\right) & : 0 \leq n \leq N-1 \\ 0 & : \text{otherwise} \end{cases} \quad (3.4)$$

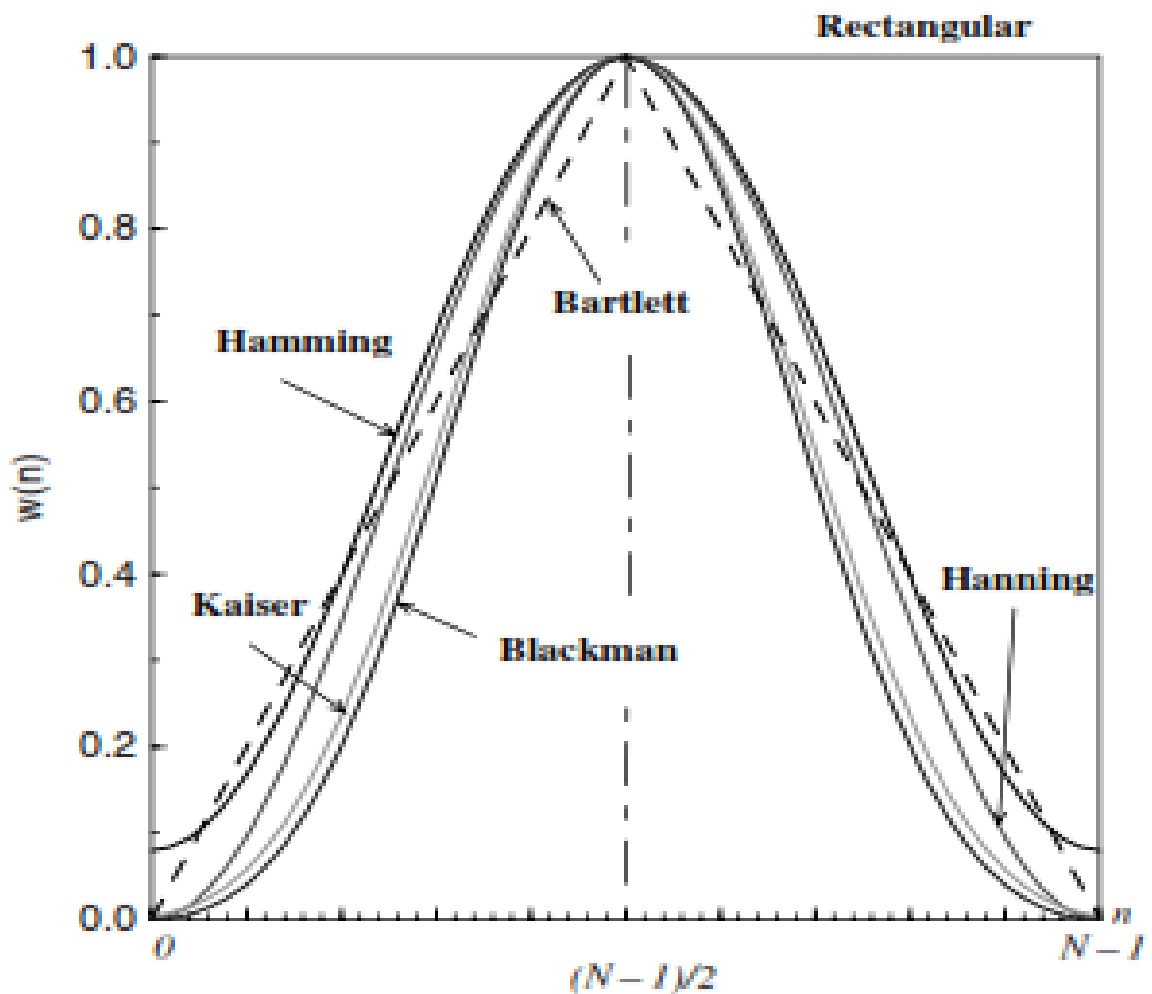


Figure 3. Time plots of various window functions

The time and frequency domain shapes of these window functions are illustrated in Figure 3, as can be seen in Figure 4, the rectangular window has the highest frequency resolution, as it has the narrowest main lobe, but the largest frequency leakage.

On the other hand, the Blackman window has the lowest resolution and the smallest frequency leakage. The effect of these windows on the time-dependent Fourier representation of speech can be illustrated by discussing the properties of two representative windows, e.g. the rectangular window and the Hamming window.

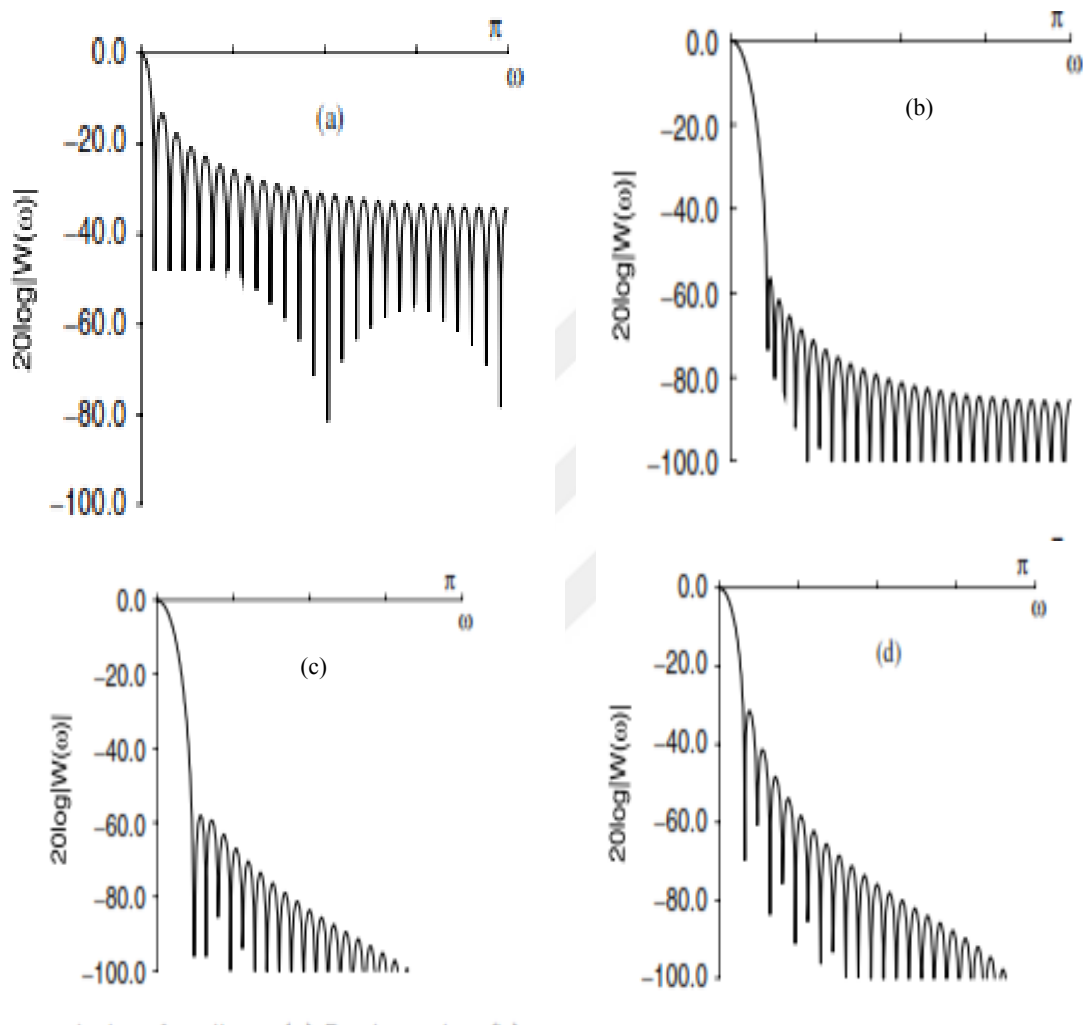


Figure 4. frequency response of various window functions: (a) Rectangular, (b) Hamming, (c) Blackman, (d) Hannin

3.1.1 Window length

Optimum window length depends on what you want to look at i.e.

- a) If we want to resolve F_0 (may be for pitch detection), the main lobe width ($2 \times W_c$) Should not exceed F_0 . For example, for hamming window,

$$\frac{8\pi}{L} \leq \frac{2\pi F_0}{F_s} \quad (3.5)$$

- b) If we want to resolve the formants, main lobe should be wide enough to “smooth together” the pitch harmonics, but not so wide that it blurs the formant peaks.
- c) If we want to look at rapidly changing events (e.g. stop release), L should be as short as possible (5-10ms max).

After windowing technique then LPC analysis is implemented. The LPC order and gain are the two main factors important for analysis.

3.2 Linear predictive coding (LPC)

LPC is used to encode analog signals where the value of a sample can be determined based on previous ones within a speech signal. The method is the work of the Department of defense in federal standard 1015, USA, published in 1984. Within ordinary settings, we sample speech at 8,000 samples per second using 8 bits for each, thus reaching 64,000 bits per second. Through LPC application, this rate can drop to 2,400 bits per second.

The present thesis further reduces the value to 1000 bits per second, at which point there might be an insignificant degree of loss as regards signal quality, rendering the method as a lossy compression approach [12-15].

In addition, the source filter model applied in LPC is called a linear predictive coding approach [17], comprising two separate segments: analysis or encoding, and synthesis or decoding. In the former stage, the signal is detected and separated to form segments known as frames, each of which is searched to further determine:

- If it is voiced or unvoiced.
- The pitch of each frame.
- Parameters required to form a filter to simulate the vocal tract.

The parameters stated above are sent to the receiver to decode and form a source filter model. This model, once supplied with the right input, can correctly regenerate the initial signal.

LPC encoding of every frame is a decision task to find out about the voiced or unvoiced nature of the frame: if voiced, an impulse train introduces it using non-zero taps that take place at pitch term intervals.

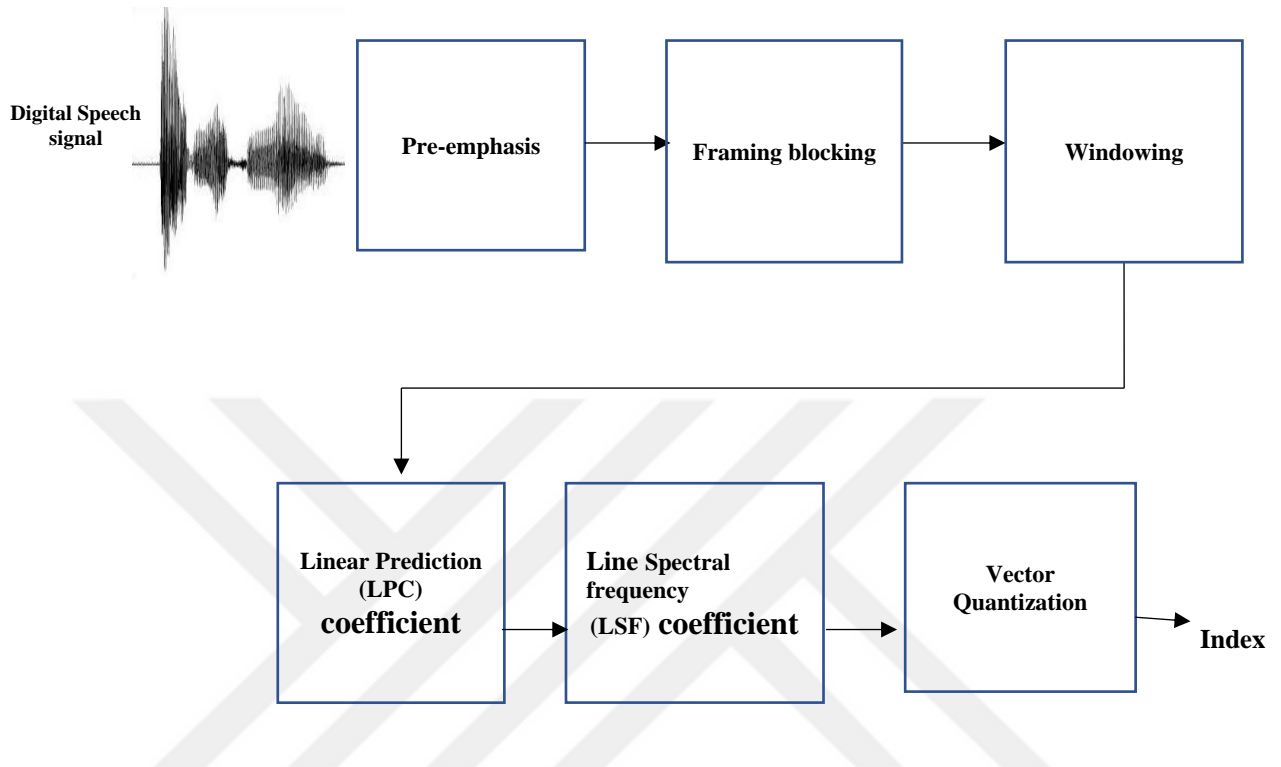


Figure 5. Block diagram of LSF coefficient calculation and quantization in speech analysis part

In the present thesis study, we apply the same approach to determine the pitch period, namely the autocorrelation technique. Should the frame be unvoiced, then it is introduced with white distortion with a 0 pitch period since the vocal cords in this situation do not tend to vibrate. As a result, the excitation to the LPC filter is either an impulse train or white distortion. Let us keep in mind that pitch, gain and filter coefficients all change according to time and each individual frame [22].

3.2.1 The LPC order definition

When we talk about LPC [77], we must be dealing with speech codecs and speech codec is nothing but just modeling of filter. For an excellent filter

1. Pass band ripple should be tending to zero.
2. Stop band attenuation should tend to infinite.
3. Transition band of Roll off.

First two things can take care of by proper windowing, that's why generally we use Hamming or Hanning window which gives good roll off also. This Roll off is the parameter which decide the no. LPC coefficient and we have a relation that

$$\text{No. of LPC coefficient (m)} = 4/\text{transition bandwidth (kHz)} \quad (3.6)$$

$$\text{and transition bandwidth} = \text{sampling frequency} * \text{some fraction} \quad (3.7)$$

This fraction must lie between 0- 0.5. So, for good roll of fraction is 0.05 no. LPC coefficient (m) will come to $4/(8\text{KHz}) * .05 = 10$. we can choose fraction less also, but it will increase the LPC coefficient which will increase the Computation time so it's a trade of between Quality and computation time. We can decrease the no. of coefficient also if we can design window, which gives roll off better than hamming window that with less no. of coefficient, we can design same quality filter.

3.2.2 Calculation of the LPC gain

The LPC excitation is defined by the formula

$$s(n) = \sum_{k=1}^p a_p(k).s(n-k) + Gu(n) \quad (3.8)$$

The LPC error is defined by the formula

$$e(n) = s(n) - \sum_{k=1}^p a_p(k).s(n-k) = Gu(n) \quad (3.9)$$

If we define

$$\sum_{k=1}^{N-1} u^2(n) = 1 \quad (3.10)$$

Then

$$G^2 = \frac{\sum e^2(n)}{\sum u^2(n)} \quad (3.11)$$

3.2.3 Calculation of LPC coefficients

In the classical least squares method, the LPC coefficients [78] are determined by minimizing the mean energy of the residual signal, given by:

$$\epsilon_p = \sum_{-\infty}^{\infty} e^2(n) = \sum_{-\infty}^{\infty} [S(n) + \sum_{k=1}^p a_p(k) \cdot s(n-k)]^2 \quad (3.12)$$

the summation range is limited by windowing either the speech or the residual signal, leading to the autocorrelation or covariance method, respectively, the autocorrelation method is computationally more efficient than the covariance method and the resulting synthesis filter is always stable.

3.2.4 Conversion of LPC to LSF coefficient and quantization

In low bit rate speech coding, the LPC coefficients are widely used to encode spectral envelope. In forward based LPC coders, the LPC coefficients are calculated from the original speech input, quantized and transmitted frame-wise. The transmission of these coefficients has a major contribution to the overall bit rate, thus it is important to quantize the LPC coefficients as few bits as possible without introducing excessive spectral distortion and with reasonable complexity, a very important requirement is that the all-pole synthesis filter $H_p(Z)$ remains stable after quantization [78].

3.2.4.1 Interpolation of the LPC coefficients

Speech analysis mechanisms employ LPC on each single frame using updated series of parameters which are calculated, quantized and then sent within the intervals of 20 to 30ms, thus leading to major parametric transformations in the next frame and, hence, generating unanticipated transients or clicks within the regenerated signal [79]. Against this backdrop, interpolation is conducted at the receiver's point to obtain minor changes in their values. Oftentimes, this can be carried out in a linear, evenly-separated time instants known as sub frames, of which there are usually four used at a time. The process is not directly carried out on the LPC coefficients because the interpolated all-pole decoding filter may, otherwise, turn rather unsteady; indeed, such steadiness resembles the conditions faced when quantization interpolation occurs on the reflection coefficients, log area ratios, inverse sine coefficients and LSF parameters with steady filters formed in all instances.

For this reason, it is common to employ LPC representation similar to quantization, as the common belief is that LSF representations offer the highest qualities in interpolation.

3.2.4.2 All pole-zero modeling of speech

Given LPC methods and their assumption of speech systems as an all-pole filter, the accuracy in calculations might suffer if, apart from poles, there are zeros in the transfer function such as the scenario with the nasal and fricative sounds. Furthermore, noisy speech adds further zeros to the range, thus significantly impacting the functionality of the all-pole LPC method. In tackling this problem, a logical solution can be to expand the model into a pole-zero form, which may be a challenge considering parameters calling for the solution of nonlinear equations first; yet, certain effective, though sub-optimal, ways are at hand to calculate these parameters hence, making them also applicable to pole-zero model the nasal and fricative sounds. Again, as regards noisy speech necessitating the assessment of just the all-pole section of the model, certain approaches are available making use of low- as well as high-order Yule-Walker formulas to determine the LPC coefficients. Reports consent to developments gained with these techniques, though not yet proven in case of extensive databases, which are complicated and may not ensure stability in the $H(z)$ values. A separate approach for gaining proper estimations of parameters for noisy signals involves multi-taper analysis, using a plurality of (orthogonal) windows, each of which offers a separate assessment of the autocorrelation coefficients or LPC parameters. Such autonomous values may later become rounded to offer accurate estimates, figure 6 shows poles and zeros.

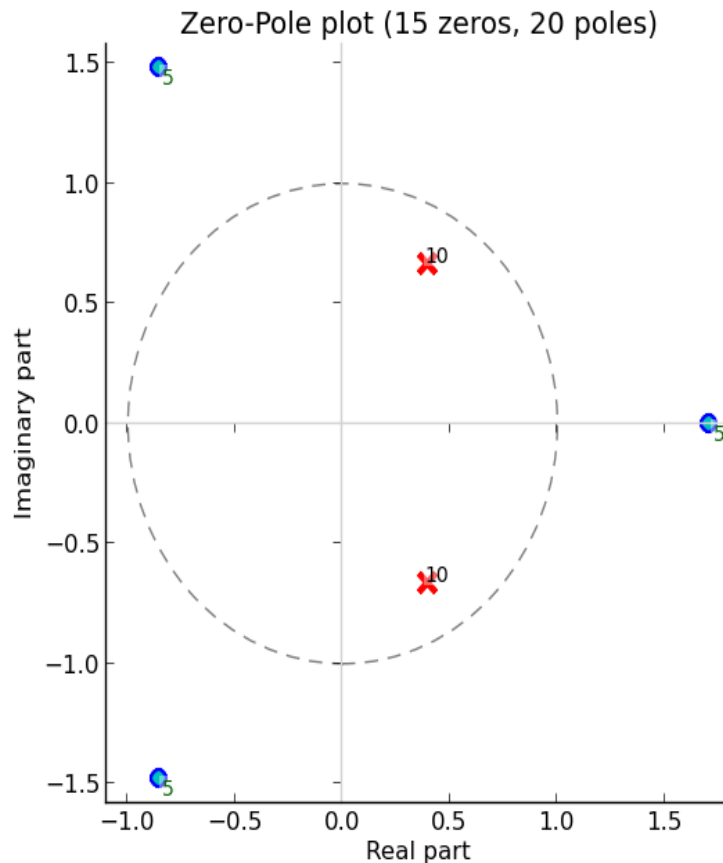


Figure 6. Poles and Zeros

3.2.4.3 Correlation of linear predictive coding

The correlations within a signal can be classified as: 1) along lags of less than 2 ms, otherwise known as short-term correlations; and 2) as a product of signal periodicity detected a long time lags of 2 ms or higher and referred to as long-term correlations [77]. The former decides the envelope of the power spectrum, whereas the latter defines the detailed configuration of the power spectrum. Both correlations may be described as redundancies while it is good to determine and analyze them initially for later signal encoding. The LPC method observes short-term correlations and defines them as LPC parameters. These are, within each targeted segment, a function of the shape of the vocal tract, whose rate of transformation is finite and, based on observations, an update rate around 50 Hz can make up for coding requirements. In this respect, linear estimation stands crucial to eliminate signal redundancy as it assesses the available amplitude in accordance to a linear mixture of previous ones [79].

Let $s(n)$ represent the sequence of samples and a_k the k^{th} predictor coefficient in a predictor of order p . Then, the sequence $\hat{s}(n)$ may be determined using

$$\begin{aligned}\hat{s}(n) &= a_1s(n-1) + a_2s(n-2) + \dots + a_ps(n-p) \\ &= \sum_{k=1}^p a_k s(n-k)\end{aligned}\tag{3.13}$$

The prediction error $e(n)$ is found from

$$e(n) = s(n) - \hat{s}(n)\tag{3.14}$$

Properly applying linear prediction calls for a time-varying filter, available sometimes by means a rearrangement of the filter once in each frame so that one can follow the time-alternating features related to speech data related to time-alternating vocal tract shape special to repeated and unique sounds. By applying such a so-called “quasi-stationary” theorem, routine speech coders can incorporate a frame size of about 20 ms for 160 samples at an 8 kHz sampling rate.

3.2.8 LPC analysis filter

Linear predictive coding is widely used in different speech processing applications for representing the envelope of the short-term power spectrum of speech. In LPC analysis [80] of order p , the current speech sample $s(n)$ is predicted by a linear combination of p past samples k , $\hat{s}(n)$

$$\hat{s}(n) = \sum_{k=1}^p a_p(k) \cdot s(n-k)\tag{3.15}$$

Where $\hat{s}(n)$ is the predictor signal and $\{a_p(1), \dots, a_p(p)\}$ are the LPC coefficients the value $\hat{s}(n)$ is subtracted from $s(n)$, giving the residual signal $e(n)$, with reduced variance

$$e(n) = s(n) - \hat{s}(n) = s(n) - \sum_{k=1}^p a_p(k) \cdot s(n-k)\tag{3.16}$$

Taking the Z-transform of equation (3.21) gives

$$E(z) = A_p(z) \cdot S(z) \quad (3.17)$$

Where $s(z)$ and $E(z)$ are the transforms of the speech signal and the residual signal respectively, and $A_p(z)$ is the LPC analysis filter of order p

$$A_p(z) = 1 - \sum_{k=1}^p a_p(k) \cdot z^{-k} \quad (3.18)$$

The filter eliminates the short-term correlation of the input signal, thus providing the output $E(z)$ having relatively a flat spectrum. Once the filter has been used, it is time for quantization of the signal, which is later decoded into a speech signal.

3.3 Line spectral frequency (LSF)

A majority of present-day speech coders employ LPC simulation for their processing, and even though some even use a backward-adaptive LPC filter [85], in most cases they obtain related parameters from the input speech at period intervals, change them to the LSF domain, and then do the quantizing for later dispatch to the decoder.

Low distortion LSF quantization plays an important role to guarantee general high standards in the synthesized signal, while the bit count designated for LSFs often takes up a majority of the total bit rate close to 50% and more in case of coders with extremely reduced bit-rate. For this reason, the general degree of efficiency in coding is predominantly related to LSF quantizer quality.

A 10th-order LPC encoding gives us an all-pole filter of 10 poles with a transfer function marked by $H(z) = 1/A(z)$, where

$$A(z) = 1 + a_1z^{-1} + \dots + a_{10}z^{-10} \quad (3.19)$$

while $[a_1, a_2, \dots, a_{10}]$ stand for LPC coefficients equally nominated by the LSF parameters as regards the zeros of a function of the polynomial $A(z)$ [76]. These parameters are determined using

$$l = [l_1, l_2, \dots, l_{10}]^T \quad (3.20)$$

Essentially, they are a scaled form of the angular frequencies otherwise called Line Spectral Pairs (LSPs) appearing between 0 and $\frac{1}{4}$. The arrangement features pertaining to the LSF parameters reveal that they are arranged and restricted within a specific scope; that is,

$$0 < l_1 < l_2 < \dots < l_{10} < 0.5 \quad (3.21)$$

On the condition that this attribute stands for quantized LSFs as well, one can state that the recreated LPC filter can remain steady. In addition, the ordering characteristic of LSF parameters [80] contains a significant part of their intra-frame dependencies. Evidently, once this characteristic is put to use properly within the quantizer, we can increase its performance manifold.

The following are the properties of LSF parameters:

- 1) All zeros of LSF polynomials are on the unit circle.
- 2) Zeros of $P(z)$ and $Q(z)$ are interlaced with each other.
- 3) The lowest phase property of $A(z)$ may be maintained without difficulty provided that the first two properties remain unchanged once quantization is complete.

3.4 Distortion Measures of quantized LSF coefficient

In order to achieve good performance quantization of LSF parameters, it is necessary to have a way of linking the quantization error to the distortion in perceptual quality. Due to the complex relationship that exists between a set of LSF coefficients and the frequency response of the corresponding LPC filter, applying a Mean-Square Error (MSE) does not necessarily lead to better performance in the system; instead, a more common approach to estimate the distortion between the original set of LSFs and the quantized version has been the Log Spectral Distortion (LSD) – though a Weighted Mean-Square Error (WMSE) might as well lead to acceptable outcomes on the condition of the right weighting function applied.

3.4.1 Spectral Distortion

The quality of quantization related to the LPC parameters is measurable using the average spectral distortion for all frames described as the root mean squared error between the power spectral density estimate of the original and the

recreated LPC parameter vector and, in more detail, based on the equation that follows:

$$D_i^2 = \frac{1}{F_s} \int_0^{F_s} [10\log_{10}(p_i(f)) - 10\log_{10}(\hat{p}_i(f))]^2 df \quad (3.22)$$

in which F_s represents the sampling frequency in Hz, and $P_i(f)$ and $\hat{P}_i(f)$ stand for the LPC power spectra of the i -th frame determined using

$$p_i(f) = 1 / |A_i(\exp(j2\pi f) / F_s)|^2 \quad (3.23)$$

and

$$\hat{p}_i(f) = 1 / |\hat{A}_i(\exp(j2\pi f) / F_s)|^2 \quad (3.24)$$

Where $P_i(z)$ and $\hat{P}_i(z)$ are the original (un quantized) and quantized LPC polynomials, respectively, for the i -th frame. The spectral distortion can be gained for all frames within the experiment to determine the average which shows the distortion related to any specific quantizer. Once again, the average spectral distortion is commonly applied to estimate the performance of quantizers.

Transparent coding means that the coded speech is indistinguishable from the original speech through listening tests. The conditions for transparent coding of speech from LPC parameter quantization are:

- 1) The average spectral distortion (SD) is approximately 1dB.
- 2) There is no outlier frame having more than 4dB of spectral distortion.
- 3) Less than 2% of outlier frames are within the range of 2-4 dB.

CHAPTER 4

DESIGN OF CODEBOOKS IN VECTOR QUANTIZATION

4.1 Quantization Theory

Quantization divides a value into a discrete number of portions generally regarded as integral multiples of the same value. A good example is rounding off, initially dealt with by Sheppard to estimate densities in accordance to histograms. Any real number 'x' may be rounded off to the closest integer – for instance, $Q(x)$ – having a quantization error of $e = Q(x) - x$.

Broadly speaking, a quantizer comprises a series of intervals or cells $s = \{S_i, i \in L\}$, where the index set L commonly represents a group of succeeding integers that start with 0 or 1, along with a series of reproduction values, points, or levels $C = \{y_i, i \in L\}$, in a way that the quantizer as a whole may be elaborated using $Q(x) = y_i$ for $x \in S_i$, written precisely as follows:

$$q(x) = \sum_i Y_i 1_{S_i}(x) \quad (4.1)$$

In which the indicator function $1_{S_i}(x)$ stands for 1 if $x \in S_i$; otherwise, it is 0. The two forms of quantizers are:

Non-uniform quantizer is the one in which the difference between the quantization levels is not uniform as shown in figure 7.

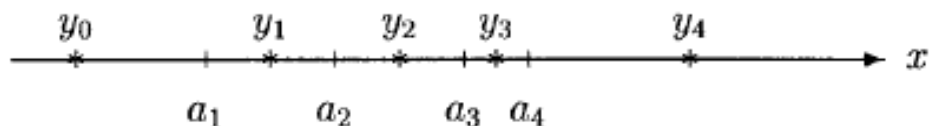


Figure 7. A non-uniform quantizer

$$a_0 = -\infty, a_5 = \infty.$$

Uniform quantizer: It is the one in which the difference between the quantization levels is uniform as shown in figure 8.

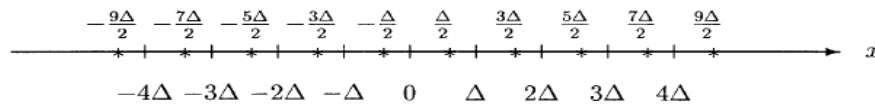


Figure 8. uniform quantizer

There are mainly two types of quantization.

- 1) Scalar Quantization
- 2) Vector Quantization

4.1.1 Scalar Quantization

Quantization being an important function for coding, quantizes each separate sample in accordance to its probability density function. An N-level scalar quantizer is regarded as a 1D form of mapping for the input range R onto an index in a codebook C, giving

$$Q : R \rightarrow C \quad C \subset R \quad (4.2)$$

The receiver applies the index to regenerate an approximation to the input level. Optimal scalar quantizers are, then, coordinated with the samples' distribution, be they determined or not in prior; if undetermined and unknown, one can make an empirical selection such as a Gaussian or Laplacian distribution) to arrange the scalar quantizer [81].

4.1.2 Scalar Quantization of LPC parameters

There are numerous scalar quantization [81] methods as per the literature on the LPC parameters, which are to be dealt with independently by applying (non)uniform systems. A non-uniform quantizer provides more reduced distortion compared to a uniform version, and it is developed based on the training dataset of the (TIMIT) database with the Lloyd algorithm – one similar to the Linde-Buzo-Gray (LBG) version for single parameters. Next, the designed quantizers become assessed as per the dataset based on spectral distortion as the criteria. Speech-coding systems need to quantize these parameters with the lowest distortion; what's more, the all-pole filter is

stable once quantization is complete. Directly quantizing the coefficients, in general, is rare because the least significant error may result in the most outstanding spectral errors, not to mention unsteady status in the all-pole filter $H(z)$. Therefore, there is no need for too many bits to carry out transparent quantization of the LPC coefficients $\{a_n\}$ per se; a 6 bits/coefficient/frame or 60 bits per frame is sufficient in the (TIMIT) database. Due to these issues, transforming is necessary for the LPC coefficients to alternative representations, thus guaranteeing a steady status for the all-pole filter once the process is complete. On top of this, such representations require one-to-one mapping – that is to say, changing representations while preserving all data related to the all-pole filter. According to previous studies, many similar representations are available among them, reflection coefficient (RC), arcsine reflection coefficient (ASRC), log-area ratio (LAR), and line spectral frequency (LSF). Here, the scalar quantization of LPC parameters is examined concerning such representations.

4.1.3 Scalar quantization using the LSF representation

LSF representation initially came to be by Itakura [82], and it possesses numerous attributes, among them bounded range, sequential ordering for parameters and an easy test of filter durability – all rendering the system very effective for quantization purposes. Furthermore, the LSF representation is of frequency-domain nature, which makes it applicable for using specific features related to human perception. For the purpose of definition, the inverse filter polynomial is applied to generate two polynomials,

$$P(z) = A(z) + z^{-(M+1)}A(z^{-1}) \quad (4.3)$$

and

$$Q(z) = A(z) - z^{-(M+1)}A(z^{-1}) \quad (4.4)$$

The roots of the polynomials $P(z)$ and $Q(z)$ are called the LSFs. The polynomials $P(z)$ and $Q(z)$ have the following two properties:

- i) All zeros of $P(z)$ and $Q(z)$ lie on the unit circle,
- ii) Zeros of $P(z)$ and $Q(z)$ are interlaced with each other; i.e., the LSFs are in ascending order.

$A(z)$ can be illustrated as the lowest stage on the condition that its LSFs meet the above-stated terms. Henceforth, the steady status of LPC synthesis filter a major requisite for speech coding – may be guaranteed without any difficulty via LPC parameters' quantization in the LSF domain.

A cluster of 2 or 3 LSFs represents a formant frequency, whose bandwidth relies upon the proximity of the respective LSFs. The spectral sensitivities of LSFs are localized – in other words, any modification within any LSF may generate a modification in the LPC power spectrum only within its neighborhood. Interpreting LSFs as formants, then, renders them appropriate to make use of specific features of the human auditory system for LPC quantization. The localized spectral-sensitivity of LSFs also causes them to become a good choice for scalar quantization since each LSF may then be quantized separately and free of any major distortion penetrated or leaked from one region to another.

4.2 Vectoral Quantization

Vector quantization [83] is a process whereby the elements of a vector of k signal samples are jointly quantized. Vector quantization is more efficient than scalar quantization (in terms of error at a given bit rate) by accounting for the linear as well as non-linear interdependencies of the signal samples. The key feature in here is a codebook C of size $N \times k$ to map the k -dimensional space R^k onto the reproduction vectors – otherwise known as code vectors or code words:

$$Q : R^k \rightarrow C, \quad C = (Y_1, Y_2, \dots, Y_N)^T, \quad Y_i \in R^k \quad (4.5)$$

This codebook resembles a limited set of vectors, $y_i: i = 1, 2, \dots, N$, chosen in advance by clustering or training so as to create the training data. While coding the vector quantization, the input samples are dealt with as blocks of k samples to generate a vector x . The VQ encoder, then, monitors the codebook and looks for an entry y_i as the closest fit or approximation for the available input vector X_t at time t . Conventional VQ reduces the distortion D to generate the optimal estimated vector X_t

$$X_t = \min_{Y_i \in C} D(X_t, Y_i) \quad (4.6)$$

The value is called the nearest neighbor encoding. As such, the specific index i generated here is the VQ representation of x . The index is designated to the chosen code vector and sent to the receiver to be reconstructed. It is important to remember that similar versions of the codebook C are to be positioned the transmitter as well as the receiver, which simply carries out a table lookup to make a quantized version of the input vector. The code rate or the rate of a vector (as it is often referred to) quantizer in bits per component can, then, be determined as in 4.7.

$$r = \frac{\log_2 N}{K} \quad (4.7)$$

Which calculates the bits per vector component applied to introduce the input vector and highlight the degree of obtainable precision with the vector quantizer on the condition that the codebook is well arranged. Given $N = 2^{rk}$, not only the encoding search complexity but also codebook storage can expand significantly based on dimension k and rate r . VQ training calls for a thorough mixture of source material to generate codebooks stable enough for quantization of data that has not been introduced in the training set. Samples of specific settings that can further supplement this set are the use of different microphones, acoustic background distortion, languages and gender. As a whole, extensive and various training sets tend to make for a relatively stable and solid codebook; yet, there is obviously no certain assurance that some unanticipated applications may still be required. A clear restriction is that algorithms used for codebooks – the Generalized Lloyd Algorithm (GLA), for example it offer merely local optimization; more updated versions like deterministic annealing and genetic optimization have, though, been introduced to tackle such disadvantages, but require more computation capacity. VQs are simply an approximate, and the hypothesis is identical to the so-called "rounding off" activity for example, toward the closest integer, For a 1D VQ illustration, see Figure 9.

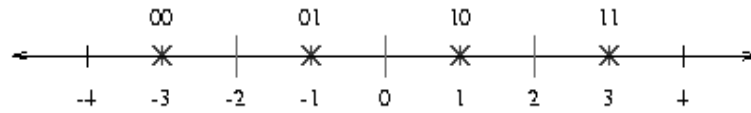


Figure 9. 1-dimensional Vector Quantizer

Where all values lower than -2 are approximated by -3; all those between -2 and 0 are approximated by -1; all those between 0 and 2 are approximated by +1; and all figures larger than 2 are approximated by +3. Let us remember that the approximate values are uniquely represented by 2 bits as a 1D VQ with a rate of 2 bits per dimension.

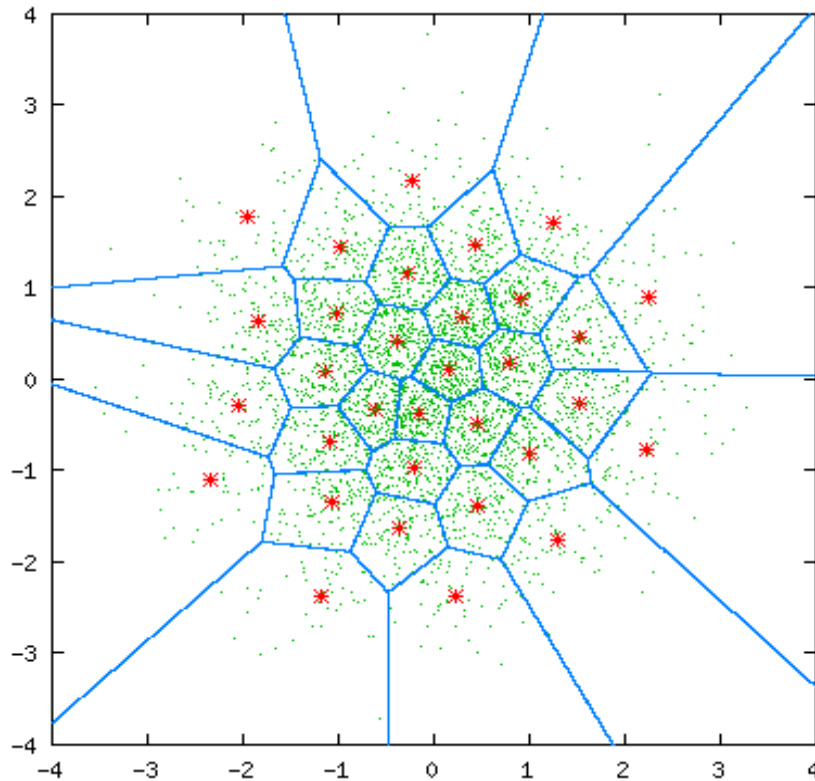


Figure 10. 2-dimensional Vector Quantizer

An example of a 2-dimensional VQ is shown in Figure 10. Here, every pair of numbers falling in a particular region is approximated by a red star associated with that region. Note that there are 16 regions and 16 red stars each of which can be uniquely represented by 4 bits. Thus, this is a 2-dimensional, 4-bit VQ.

Its rate is also 2 bits/dimension. In the above two examples, the stars are called code vectors and the regions defined by the blue borders are called encoding regions. The set of all code vectors is called the codebook [84] and the set of all encoding regions is called the partition of the space.

4.2.1 Design of codebooks

The VQ design problem [84] can be stated as follows. Given a vector source with its statistical properties known, given a distortion measure, and given the number of code vectors, find a codebook (the set of all red stars) and a partition (the set of blue lines) which result in the smallest average distortion. We assume that there is a training sequence consisting of M source vectors:

$$T = \{X_1, X_2, \dots, X_M\} \quad (4.8)$$

This training sequence can be obtained from some large database. For example, if the source is a speech signal, then the training sequence can be obtained by recording several long telephone conversations. M is assumed to be sufficiently large so that all the statistical properties of the source are captured by the training sequence. We assume that the source vectors are K -dimensional, e.g.,

$$X_m = (X_{(m,1)}, X_{(m,2)}, \dots, X_{(m,k)}) \quad m=1,2,3,\dots,M \quad (4.9)$$

Let N be the number of code vectors and let

$$C = \{c_1, c_2, \dots, c_N\} \quad (4.10)$$

Represents the codebook. Each code vector is K -dimensional, e.g.,

$$C_n = \{c_{(n,1)}, c_{(n,2)}, \dots, c_{(n,k)}\} \quad n=1,2,\dots,N \quad (4.11)$$

Let S_n be the encoding region associated with code vector C_n and let

$$p = \{s_1, s_2, \dots, s_N\} \quad (4.12)$$

denote the partition of the space. If the source vector X_m is in the encoding region S_n , then its approximation (denoted by $Q(X_m)$) is C_n :

$$Q(X_m) = C_n \quad \text{if } X_m \in S_n \quad (4.13)$$

Assuming a squared-error distortion measure, the average distortion is given by:

$$D_{ave} = \frac{1}{MK} \sum_{m=1}^M \|X_m - Q(X_m)\|^2 \quad (4.14)$$

Where $\|e\|^2 = e_1^2 + e_2^2 + \dots + e_k^2$ the design problem can be succinctly stated as follows: Given T and N , find C and P such that D_{ave} is minimized.

4.2.2 Optimality Criteria

If C and P are a solution to the above minimization problem, then it must satisfy the following two criteria.

4.2.3 Nearest Neighbor Condition

$$S_n = \{X : \|X - C_n\|^2 \leq \|X - C_{n'}\|^2 \quad \forall n' = 1, 2, \dots, N\} \quad (4.15)$$

The condition says that the encoding region S_n should consist of all vectors that are closer to C_n than any of the other code vectors. For those vectors lying on the boundary any tie breaking procedure will do.

4.2.4 Centroid condition

$$C_n = \frac{\sum_{x_m \in S_n} X_m}{\sum_{x_m \in S_n} 1} \quad (4.16)$$

This condition says that the code vector C_n should be average of all those training vectors that are in encoding region S_n . In implementation, one should ensure that at least one training vector belongs to each encoding region (so that the denominator in the above equation is never 0).

4.3 Codebook Generation Algorithms

The quality of vector-based quantization may surpass that of scalar-based ones with reduced bit-rates; still, this benefit comes with major computation and storage spending. To make up for this loss, numerous kinds of codebooks are proposed – some pre-computed and remaining unchanged as they are employed. In other cases, they may be brought up to date while quantizing is taking place. at this point of the study, we elaborate on the most popular codebooks.

4.3.1 Full Search Codebook

A full search codebook is one where during the quantization process each input vector is compared against all of the candidate vectors in the codebook. This process is called full search or exhaustive search. The computation and storage necessary for usual full search codebooks works in the following way: should every vector be represented by $B = RN$ bits for transmission, the vector count can be estimated using

$$L = 2^B = 2^{RN} \quad (4.17)$$

in which, N represents vector size. A number of these applications make the assessment of absolute values for error an unnecessary task since the prime target is the choice of the most ideal vector. Accordingly, an approximate performance and not an evident or absolute error – will be sought instead; hence, the chance to measure the resemblance instead of the difference between the input vector and those in the codebook. In other words, computing the cross-correlation of the input vector with each of the options in the codebook dates, those with the best cross-correlation figures

are chosen as the quantized value of the input vector. As for the computation cost on the condition that all the vectors are normalized, the differences in the energy levels may provide conflicting cross-correlation values and, hence, the cost may be determined by,

$$com_{f_s} = N2^{RN} \text{ multiply – add per input vector} \quad (4.18)$$

Based on which, one may as well determine the storage values for the codebook vectors as,

$$M_{f_s} = NL = N2^B = N2^{RN} \text{ locations} \quad (4.19)$$

As visible from the expression, these two requirements are predominantly based on the bit count in the code words.

4.3.2 Binary Search Codebook

Binary search, otherwise referred to in pattern recognition as hierarchical clustering, simply partitions the distance so as to proportionate the search for lowest distortion code-vector with $\log_2 L$ instead of L . Within the related discipline, binary search codebooks may as well be named tree codebooks or tree search codebooks [14],[17]. In a binary search codebook, N dimensional space is initially separated to form two areas with the help of the K-means algorithm and two primary vectors; next, each area is additionally subdivided to form two sub-areas the process continuing as such. Up to the point where the space is divided into L areas, regions, or cells to put it differently. At this point, L becomes restricted to represent a power,

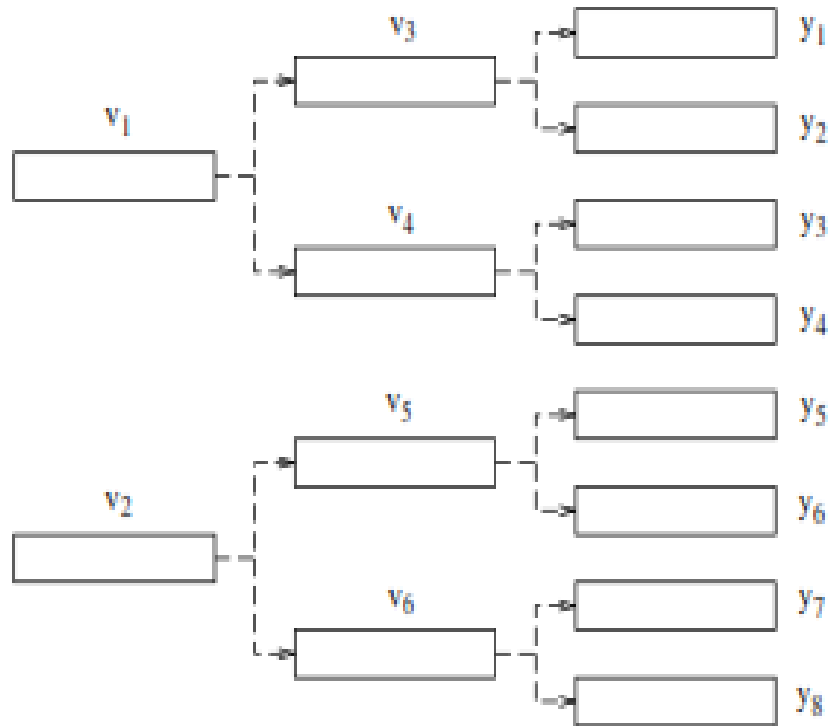


Figure 11. Binary splitting into eight cells

where B represents an integer number of bits. Each area or region can be related to one centroid. Figure 11 represents such space division to make up $L = 8$ cells. Within the initial binary split, v_1 and v_2 can be determined as the centroids of the two halves of the entire space under operation. The next binary split calculates four centroids. The centroids of the regions after the third binary division are the actual codebook vectors y_i an input vector \mathbf{x} is quantized by examining the tree along a direction with the least distortion at each node. Once more, given N multiply–adds to compute each distortion, the related cost is determined as follows:

$$C_{\text{mbs}} = 2N \log_2 L = 2NB \quad (4.20)$$

Within each step, the input vector is matched as opposed to just two alternatives or candidates, thereby turning the computation cost into a linear function of the bit count within the code words. The overall storage cost, nevertheless, spikes considerably in this way as

$$M_{\text{bs}} = 2N(L-1) \quad (4.21)$$

A tree search codebook does not necessarily call for a binary search codebook and, oftentimes, division steps can be less than the bit count, B , in the code word.

Under such circumstances, every vector from the former step can indicate more than two vectors in a current step.

4.3.3 LBG Algorithm

The LBG VQ design algorithm [85] is an iterative algorithm which alternatively solves the above two optimality criteria. The algorithm requires an initial codebook $C^{(0)}$. This initial codebook is obtained by the splitting method. In this method, an initial code vector is set as the average of the entire training sequence. This code vector is then split into two. The iterative algorithm is run with these two vectors as the initial codebook. The final two code vectors are split into four and the process is repeated until the desired number of code vectors is obtained. The algorithm is summarized below.

4.3.3.1 LBG Algorithm design

- 1) Given T Fixed $\epsilon > 0$ to be a small number
- 2) Let $N=1$ and

$$C_1^* = \frac{1}{M} \sum_{m=1}^M X_m \quad (4.17)$$

Calculate

$$D_{ave}^* = \frac{1}{MK} \sum_{m=1}^M \|X_m - C_1^*\|^2 \quad (4.18)$$

- 3) Splitting For $I = 1, 2 \dots N$, set

$$C_1^{(0)} = (1 + \epsilon)C_1^* \quad (4.19)$$

$$C_{N+1}^{(0)} = (1 - \epsilon)C_1^* \quad (4.20)$$

Set $N = 2N$

- 4) Iteration Let $D_{ave}^0 = D_{ave}^*$. Set the iteration index $i=0$

a) For $m = 1, 2, \dots, M$, find the minimum value of

$\|X_m - C_n^i\|^2$, over all $n=1, 2, \dots, N$. Let n^* be the index which achieves the minimum. Set

$$Q(X_m) = C_{n^*}^i \quad (4.21)$$

b) For $n = 1, 2, \dots, N$, update the code vector

$$C_n^{(i+1)} = \frac{\sum_{Q(X_m)=C_n^i} X_m}{Q(X_m)=C_n^i} \quad (4.22)$$

c) Set $i=i+1$

d) Calculate

$$D_{ave}^{(i)} = \frac{1}{MK} \sum_{m=1}^M \|X_m - Q(X_m)\|^2 \quad (4.23)$$

e) If $(D_{ave}^{(i-1)} - D_{ave}^{(i)}) / D_{ave}^{(i-1)} > \epsilon$, go back to step (a)

f) Set $D_{ave}^* = D_{ave}^{(i)}$. For $n = 1, 2, \dots, N$ set $C_n^* = C_n^i$ as the final code vectors

g) Repeat steps (c) and (d) until the desired no of code vectors are obtained.

Here the LBG design algorithm is run with $\epsilon=0.001$

4.3.3.2 Flow Chart for LBG Algorithm

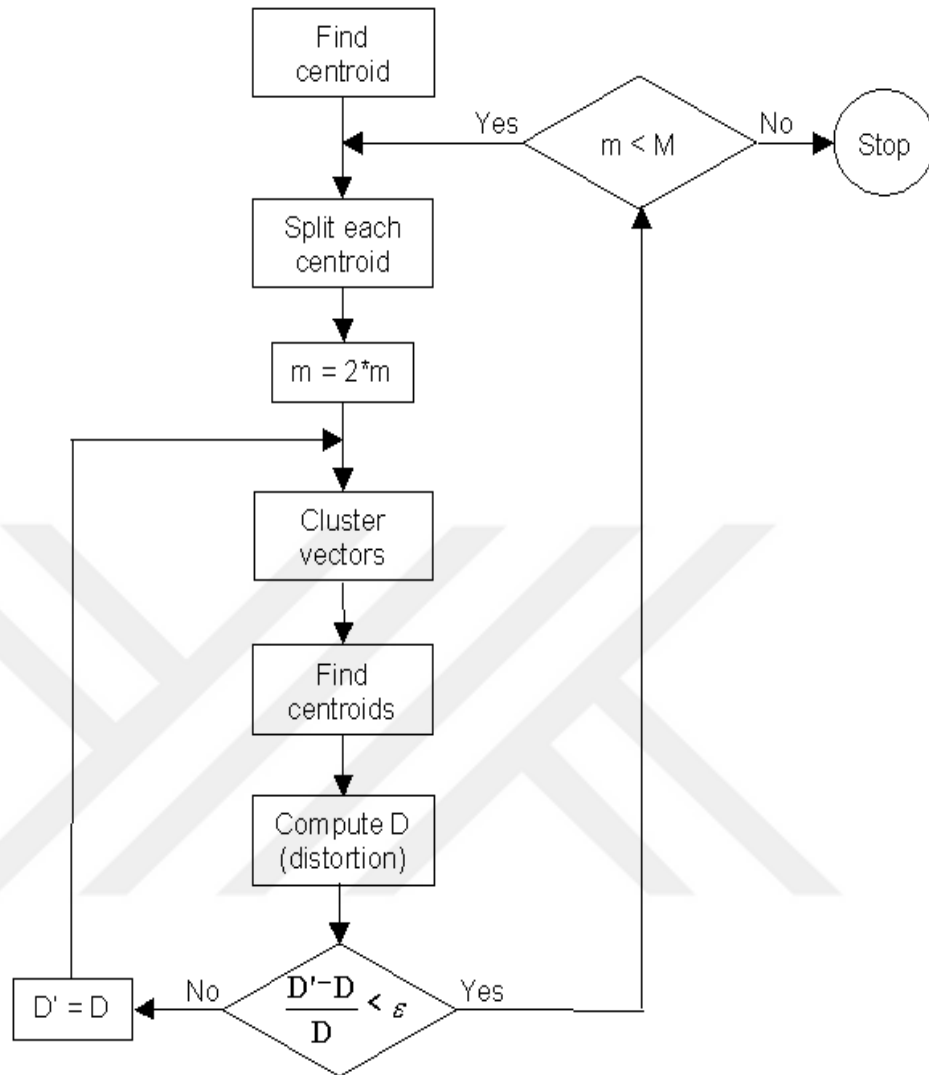


Figure 12. Flow diagram of the LBG algorithm (Adapted from Rabiner and Juang, 1993)

Intuitively, the LBG algorithm designs an M -vector codebook in stages. It starts first by designing a 1-vector codebook, then uses a splitting technique on the code words to initialize the search for a 2-vector codebook and continues the splitting process until the desired M -vector codebook is obtained. Figure 10 shows, in a flow diagram, the detailed steps of the LBG algorithm. "Cluster vectors" is the nearest-neighbor search procedure, which assigns each training vector to a cluster associated with the closest codeword. "Find centroids" is the centroid update procedure.

"Compute D (distortion)" sums the distances of all training vectors in the nearest-neighbor search so as to determine whether the Procedure has been converged

4.4 Training, Testing and Codebook Robustness

Key to codebook design is the training for populating the codebook. Basically, the procedure enhances a codebook for the assigned training data through measuring the centroids of each cell. Considering that the K-means algorithm may not always yield desired outcomes in a globally optimized codebook, sometimes it is better to re-run the algorithm using a series of alternative codebook vectors [19].

Once a codebook has been prepared to correspond to an assumed group of training data, testing is needed for its performance using data not previously included in training. This is because mere testing with the training data results in improved performance compared to what a codebook might present in real settings. The degree of strength within a codebook may be assessed by testing the performance on data with a varying distribution compared to the training set. In actual settings, it is impossible to forecast all possible scenarios for quantizer application, and the distribution of the actual data can vary from that of the training set. Here, two alternatives emerge to impact the design and operational performance of a codebook: input signal variations and digital transmission channel errors.

CHAPTER 5

VECTOR QUANTIZATION METHODS

5.1 Introduction

Quantization maps an unlimited series of scalar or vector values using a limited set of such quantities, and it is employed to process signals, speech and images. To code speech, this method decreases the bit count for sample representation; once this is accomplished, complexity and memory needs are also reduced. Quantization also causes reduced within the signal an effect that is not favorable and which calls for a solution; such a remedy can be found in balancing the bit-rate reduction and signal quality. As stated before, the two forms of quantization are scalar (SQ) and vector-based (VQ); the former is done on a sample-by-sample routine, whereas the latter is conducted using groups of vectors.

Vector quantization adds to the quality of a quantizer in the expense of a spike in computational and memory-related restrictions and expenditure. According to Shannon theory, to quantize a vector can be done with higher outcomes compared to quantizing each scalar value for spectral distortion, which implies that the selected vector dimensions highly impact the quantization quality as a whole; those of higher dimensions tend to result in more advanced quality as opposed to lower-dimension vectors as in the latter the degree of transparency often drops in certain bit-rates [8]. The reason for this lies in the correlation being lost between the samples and also the scalar quantization distorting the correlation between consecutive samples, thus leading to loss of quality in the quantized signal. For this reason, quantizing correlated data has to apply certain methods to maintain this correlation – a task carried out using vector quantization as an interpretation for scalar quantization. In this method, larger vectors generate transparency based on a certain designated bit-rate. VQ quantizes the data as adjoining blocks known as vectors instead of each sample; however, as coding

becomes more advanced, one can now obtain satisfactory transparency for all vectors, including smaller ones.

5.2 Multistage Vector Quantization (MSVQ)

There are various ways to estimate codebooks in the MSVQ approach [12], the easiest of which is training them on a successive basis. In this way, the codebook is initially computed conventionally and with, say, GLA to quantize the data using a one-step quantizer. The final error vectors are, then, applied for data training in the upcoming phase – the process continuing in the same way for the remaining phases with individual and new codebooks created based on the error between the original and the recreated vectors.

The multi-stage vector quantizer is of product-code nature and decreases the complexity factor, yet compromising the performance. In 2-stage VQ [13], the LPC parameter vector, appearing in a more appropriate form – for example, as LSF - is quantized by the first-stage vector quantizer and the error vector e as the difference between the input and output vectors of the initial stage is also quantized by the second-stage vector quantizer. The ultimate LPC vector is, then, created by adding up the outputs of these two steps. In order to reduce the complexity within the 2-stage vector quantizer, the LPC bits are shared evenly between the two stages. Key to the process is choosing the right distortion measure; as the spectral distortion is employed to assess LPC quantization, it is best to do so in devising the vector quantizer, but this is a rather challenging task; instead, more basic criteria are used – including the Euclidean and the weighted Euclidean distance values - between the main and quantized LPC parameter vectors (better presented as LSF). To determine the ideal LPC parametric representation for the Euclidean distance measure, one has to investigate a 2-stage vector quantizer with the distance measure in three domains of – namely - LSF, arcsine reflection coefficient, and log-area ratio. The quantizer operates more efficiently using LSF representations compared to the other two. The Euclidean distance measure employed for VQ processing in the previous stage results in even weights for each LSF vector that are, of course, disproportionate as to their degree of spectral sensitivity. A suggestion by Paliwal and Atal is weighted Euclidean distance measures in the LSF domain to designate individual weights based on such sensitivity criteria. The weighted Euclidean distance measure between the test LSF vector f and the reference LSF vector \hat{f} is given by:

$$d(f, \hat{f}) = \sum_{i=1}^{10} [C_i W_i (f_i - \hat{f}_i)]^2 \quad (5.1)$$

Where f_i and \hat{f}_i are the i -th LSFs in the test and reference vector, respectively, and C_i and w_i are the weights assigned to the i -th LSF. These are given by

$$C_i = \begin{cases} 1.0 & \text{for } 1 \leq i \leq 8 \\ 0.8 & \text{for } i=9 \\ 0.4 & \text{for } i=10 \end{cases} \quad (5.2)$$

and

$$W_i = [P(f_i)]^r \quad \text{for } 1 \leq i \leq 10 \quad (5.3)$$

where $P(f)$ stands for the LPC power spectrum related to the test vector as a function of frequency f , and r represents an empirical constant governing the relative weights assigned to various LSFs and determined on an experimental basis. A value of r equal to 0.15 has been seen to give the highest performance. The 3-stage vector quantizer with the weighted LSF distance measure calls for an average 24 bits/frame to materialize transparent quantization of LPC parameters (with roughly 1dB as spectral distortion, lower than 2% outliers in the range 2-4 dB, and no outlier with spectral distortion exceeding 4 dB). The 3-step MSVQ block diagram appears in Figure 13, where x is the input vector, e_1 is the error vector and also the input for the 3rd stage. In the same way, e_2 represents the error vector for 3rd stage and also the input for the 3rd stage; hence completing the MSVQ process.

$$e_1 = x - \hat{x}_1 \quad (5.4)$$

and

$$\hat{e}_1 = Q[e_1] \quad (5.5)$$

$$\hat{x} = \hat{x}_1 + \sum_{i=1}^{m-1} \hat{e}_i \quad (5.6)$$

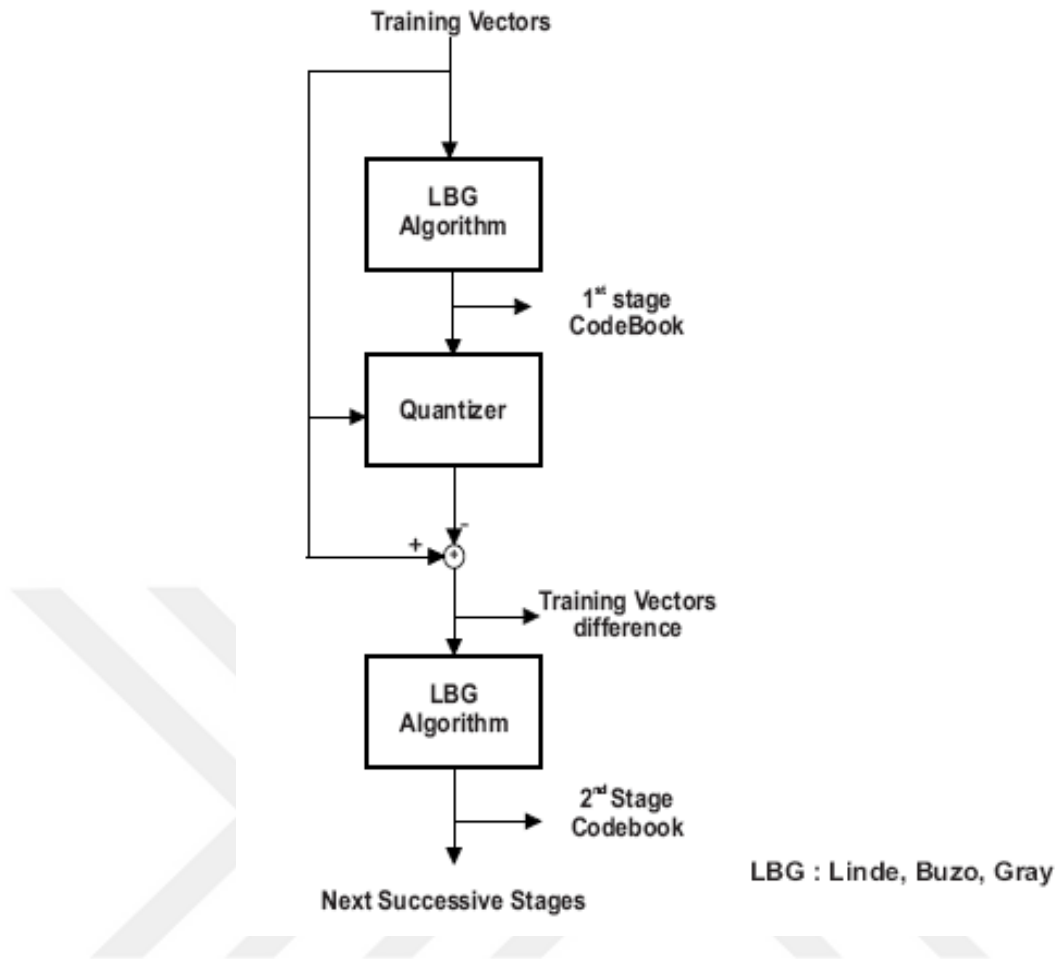


Figure 13. Codebook generation for different stages of MSVQ.

Here, m represents the number of stages and, given that each represents an independent vector quantiser, the required bits are to be assigned individually. Also, considering every quantiser performing with reduced bit rates, this in turn economizes on memory as well as computational requirements. Matching the entire number of floating-point values of the codebooks of an m stage, n dimensional MSVQ operating at b bits/vector, to that of an equal unrestricted quantiser with an identical bit rate and dimensionality, one obtains:

$$\sum_{i=1}^m n2^{b_i} \leq n2^b \quad (5.7)$$

Where b_i is the number of bits given to the i th stage vector quantiser and nevertheless, the sequential training algorithm cannot successfully use the inter-stage dependencies in the codebook optimization.

$$b = \sum_{i=1}^m b_i \quad (5.8)$$

To enhance the performance of the sequential training, we may apply an iterative algorithm otherwise considered as a mutual design of the stage codebooks [8]. Here, the error vectors are determined as the error between the original vector and the multistage reproduction vector to accommodate the entire stages excluding the present one applied for re-optimization of the codebook. Still, the concurrent and shared design algorithm put forth in [9] provides an additional stage to enhance performance and speed up the rate of convergence. The computational complexity associated with the MSVQ technique relies upon the specific search algorithm employed for analysis, and an effective search calls for distortion measures to be assessed for all probable vector configurations; in this way, there will be no difference in terms of the degree of complexity from the full-search VQ approach. Contrary to this, the easiest and yet not quite effective and optimal algorithm for MSVQ is considered to be the sequential search, whereby next stages are neglected within each current stage; in detail, past the initial stage, the residual from the former stages can be quantized separately and individually, though there can still exist more effective trade-offs between performance and complexity if better search algorithms are employed. In [6], it is asserted that the M - L search, where the M best vector configurations are sought within every one of the L stages, performance can be obtained as efficient as the optimal search with rather insignificant degrees of complexity.

5.3 Split Vector Quantization (SVQ)

The main disadvantage of Unconstrained Vector Quantizer is high complexity, memory requirements and the generation of codebook is a difficult task as vectors of full length are used for quantization without any structural constraint. As a result, a greater number of training vectors and bits cannot be used for codebook generation. Given this limitation, the quantizer is unable to generate any more optimal outputs, thus calling for another famous method known as Split Vector Quantization (SVQ).

The main idea behind it is to divide the vectors of larger dimensions into those with smaller sizes, and also to distribute the assigned bits among the so-called splits or sections. Upon splitting, the vector dimension is reduced, hence requiring additional training vectors and bits to form the codebook and, in turn, boosting quantization performance and reducing complexity and memory requirements as a whole. Despite this benefit, the drawback remains that, upon splitting, the linear and nonlinear dependencies between the samples of a vector disappear, compromising the shape of the quantizer cells and, eventually, minimally adding to spectral distortion. To make up for this increase, one may add to the training vectors and bits to create the codebook. The split count in such quantizers should be carefully restricted, or the vector quantizer will operate in a similar way to a scalar one, instead. SVQ divides the training sequence into vectors of lower size where each division or split of the training sequence can, then, be employed to form independent sub-codebooks, thereby forming independent vector quantizers with bits to be allotted to each one. The outcome is that fewer bits may be at hand at each quantizer, in turn decreasing complexity and memory requirements because they rely on the assigned bit count to the quantizer as well as on the dimension of the vector to be quantized.

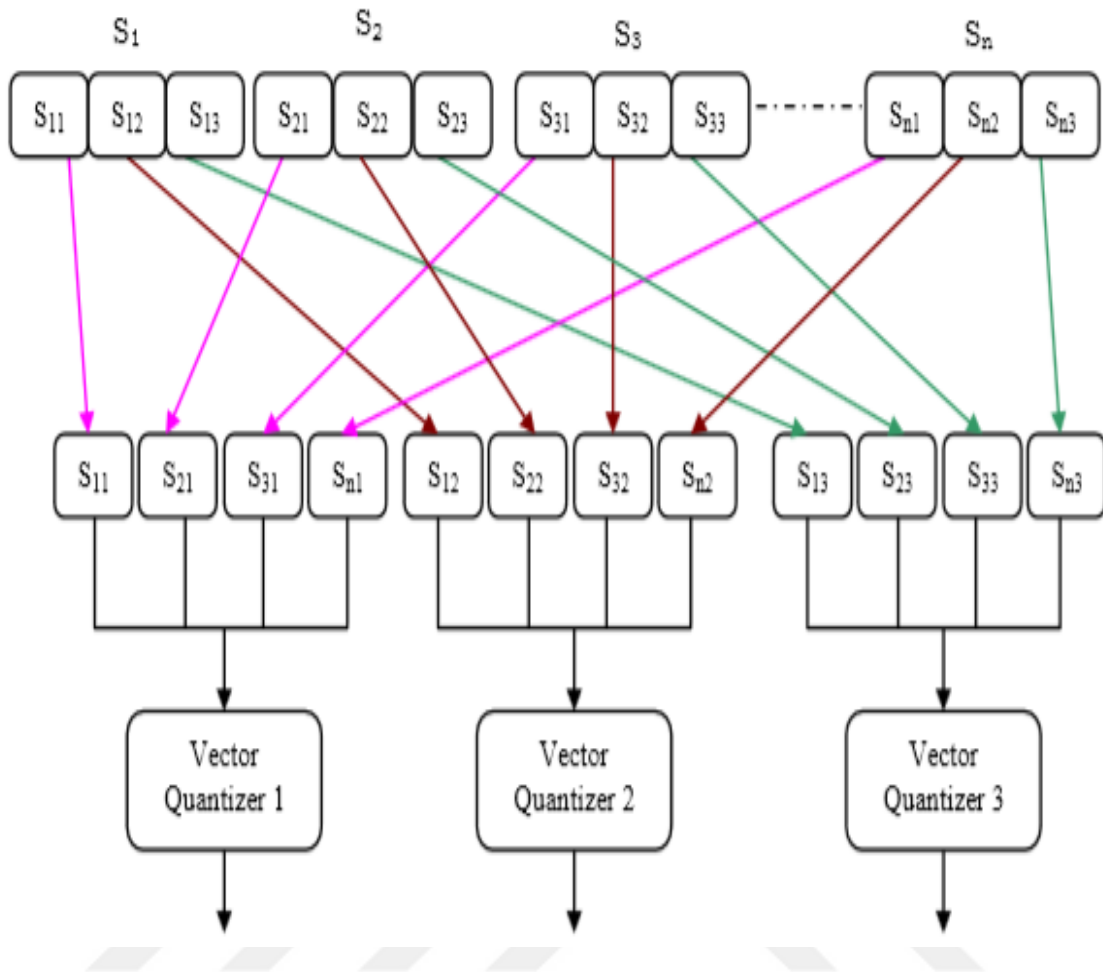


Figure 14. Block diagram of three-part split vector quantizer

Figure 14 shows the block diagram of a three-part Split vector quantizer. Accordingly, a vector S_1 of dimension ‘n’ is quantized by being divided into sub-vectors S_{11}, S_{12}, S_{13} of lower dimensions. Every one of the sub-vectors is, then, quantized based on the corresponding codebook. In the present research, the order of the filter taken is 10 and, as such, the LSF vector carries 10 samples divided into three parts of (3,3,4), (4,4,2) and (4,3,3) samples [49, 68-70].

In an ‘n’ dimensional Split Vector Quantizer of ‘sp’ splits and ‘b’ bits per vector, the vector space R^n is divided into ‘sp’ subspaces, splits, or divisions of smaller size; next, the dimension of a subspace is n_i where $n = \sum_{i=1}^{sp} n_i$. The independent quantizer count equals that of the splits, with the bits utilized for quantization distributed among the splits. With b_i as the bits assigned to each split of the vector quantizer, the entire number of bits assigned can be determined with $b = \sum_{i=1}^{sp} b_i$ [54].

5.4 Residual vector quantization (RVQ)

RVQ is a type of multistage vector quantization. RVQ is implemented with a direct-sum codebook structure, it decomposes an input vector stage-wise. This successive decomposition starts from the first stage, where an input vector is mapped to one of the codevector in the codebook of the first stage. The mapping of the input is done according to some distance criterion. The mapped codevector of the first stage is then subtracted from its input to yield a residual vector for the first stage. The residual is fed to the next stage as the input. The process continues for every subsequent p^{th} stage, and the respective residual vector is created by subtracting the mapped codevector y_p of the p^{th} stage from the input of that stage. This process stops if either the last stage P is reached, or when the MSE between the original input and the reconstructed input at a stage meets a prespecified threshold. The reconstructed vector of the original input vector is obtained by summing up the corresponding y_p codevectors of all the used staged. For all the P stages of RVQ, the reconstructed speech. The entire operation of RVQ can be summarized in the following:

- a) A mapping to direct-sum codevector: this function is mapping from R^k to R^{th} , where k is the dimensionality of the codevectors and also the input space.
- b) A mapping to P -tuple representation of the direct-sum codevectors: P -tuple is a set $\{i_1, i_2, i_3, \dots, i_p, \dots, i_p\}$ where $i_p \in \{1, 2, \dots, M\}$ is the index of one of the M codevectors at the p^{th} stage of the RVQ. This mapping is a transformation from R^k to R^P is the total number of stages of the RVQ and, generally, $P \ll K$.
- c) Mapping back from R^P to the input space R^k : the P -tuples are transformed back in to the input space to give the reconstructed speech of the input speech.

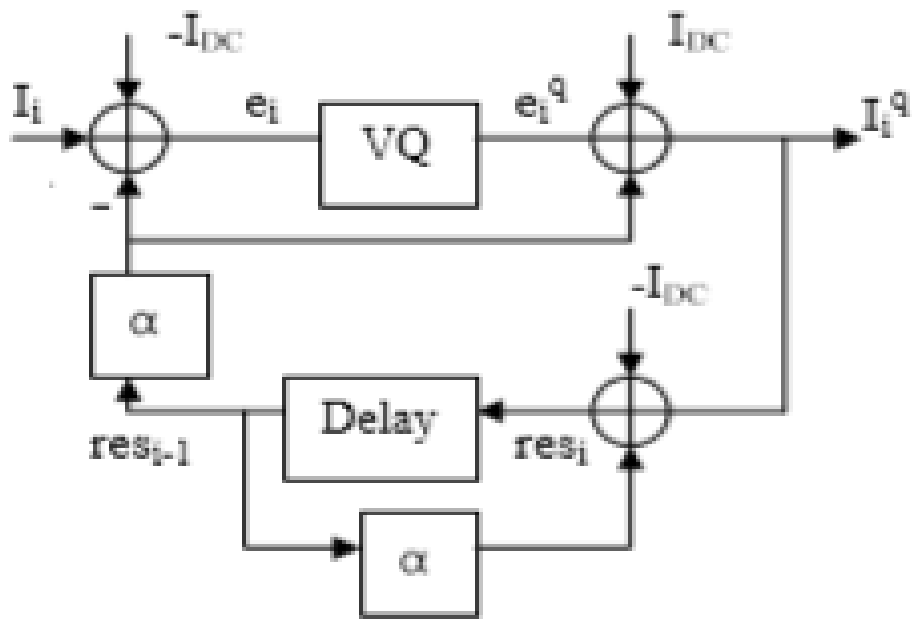


Figure 15. Residual vector quantizer

CHAPTER 6

CODEBOOK DESIGN WITH MSVQ AND SVQ METHODS

6.1 Introduction

Speech coding is a process of representation of an audio signal for efficient transformation. It is based on the form of lossy algorithms. This form of the algorithm is considered acceptable as regards to the encoding of speech since the loss in quality tends to remain undetectable by the human ear. For the speech that is uncompressed, the transmission occurs at the 64kb/s (8bits/sample) and 8 kHz for the sampling. A bit rate that appears lower than this figure is considered to be compressed. The process by which a speech signal is being transformed to a compressed format is tagged as the speech coding. This transmission is usually in the form of few binary numbers.

Linear predictive coding is a major part of the speech compression algorithm. In the practical application of the LPC coding parameters, there is no place for the direct quantization. As a result, there is a need to ensure conversion to the line spectral frequency (LSF). This process of conversion is found to ensure there is important stability in the poles. Another form of quantization is the 'split vector quantization'. This has been proposed to another choice or alternative especially in a situation where there is vector input that is divided into multiple sub-vectors. This quantization of the vector is done with a different codebook. Results found to be associated with the split vector quantization, it was noted that such a process help in the reduction of the computational complexity and the storage space required for the vector quantization. The shortcoming of such relates to it SD which is noted to be high especially when been compared to others. Based on the available information, as regards to the multistage VQ, (MSVQ) was proposed to help in the `quantize the LSF parameters especially in several follow-up stages. When a comparison was done as regards to the MSVQ and SVQ, the MSVQ shows a better performance especially in terms of

applying a transformed and compounding as the preprocessing and post-processing blocks of SQ [87-92].

Designing an optimal vector quantizer is the goal to obtain a quantizer consisting of N reproduction vectors, such that it minimizes the expected distortion. The codebook is then designed and trained by the training vectors associated with the popular design method noted to be related to the LBG algorithm. This process is a form of an iterative algorithm which tends to require the initial codebook formed from the splitting method. Based on this method that is been described, there is an initial code vector which is thus set as an average for the entire training sequences. Afterward, the code vector is then split into two while these two coded vectors is then split into four. The splitting process continues until the required amount of code vectors is attained. In encoding the real process, there is input vector which then searches for the best matching codebook vector from the obtained codebook and index which is transmitted. The function of the decoder is to work like a table look-up which receives the transmitted index and look at the codebook for the vector which corresponds to and then matched vector from the codebook is then used to represent the input vector.

In our work, the experiment is carried out using standard TIMIT speech database containing American and British English speeches, which had been sampled at 8 kHz. In order to design SVQ or MSVQ codebooks, the LSF vectors were generated from this TIMIT database. The sentences were spoken by both female and male speakers. The frame length is about 30 ms with hamming window and overlap 10ms. 10th order LPC analysis based on an autocorrelation method was performed for every 20 ms frame. The resulting coefficients of the 10th order LP polynomial $A(z)$ were converted into the LSF domain. The LSF input data was used for designing of codebooks. The input data used in the experiments consisted of two separate databases as training and test databases. The training database were consisted of 60 minutes speech, this speech is divided into clean and noisy speech for test database were consisted of clean speech about 45 minutes and 15 minutes for noisy speech. After design of codebooks, the spectral distortion (SD) values were calculated over the frequency band of 100-3800Hz for 8 kHz sampled speech.

6.2 Split Vector Quantization (SVQ) Codebook design

Direct quantization of a set of LSF parameters with a typical vector quantizer of 25 bits, would require a codebook with 1025 entries, which is not practical from both the search complexity and memory point of view. An alternative method is to use SVQ, where the 10-element LSF vector is split into a number of smaller subvectors, each quantized independently using a small number of bits. Since the complexity and storage requirements of a full-search vector quantizer are exponential functions of the number of bits used to represent the input vector, SVQ requires only a fraction of the complexity required by a full search VQ.

Splitting the 10-element LSF vector can be performed in various ways. The split usually takes into account some of the perceptual properties of the LSF vector, such as the fact that lower frequency LSFs are usually more sensitive to distortion than higher frequency ones. Therefore, a {4,6} split would be preferred to a {5,5} split for instance. The configurations shown here have been chosen so that they all have the same bit-rate of 24 bits. Complexity (in multiply-adds) and memory storage (in words) for the typical SVQ configurations are presented in Figure 20, 21. It can be seen that although the direct VQ approach is extremely complex, the SVQ configurations are all practical. Even the most complex one requires only 40960 multiply-adds per input vector, however, there are several drawbacks which relate to the efficiency of SVQ quantization:

The number of bits allocated to each subvector is fixed. The effect of the weighting function will therefore be limited to within one subvector. If a subvector contains only LSFs of relatively small importance, they will still use all the bits allocated to this subvector, whereas a classic VQ would effectively shift some of that band width towards the more important LSFs, through the weighting function. This effectively reduces the use of the weighting function to the LSF within a given subvector and lowers the overall quantization efficiency of an SVQ quantizer. 3-split codebooks were presented which are (3,3,4), (4,4,2) and (4,3,3). The most common splitting scheme is (3,3,4) in which the first split contains the first three components of the 10-dimensional LSF vector, LSF1–LSF3, the second split consist of LSF4–LSF6 while LSF7–LSF10 constitute the third split. Then, we have used (6,4), (4,6), (5,5) splitting in 2-split codebooks We have calculated the spectral distortion and memory

requirements for each designed codebooks. The spectral distortion results of 2-split & 3-split SVQ codebooks for clean speech input data can be seen in Table 1 and Table 2, respectively. The spectral distortion results of 3-split SVQ codebooks for noisy speech input data can be seen in Table 3.

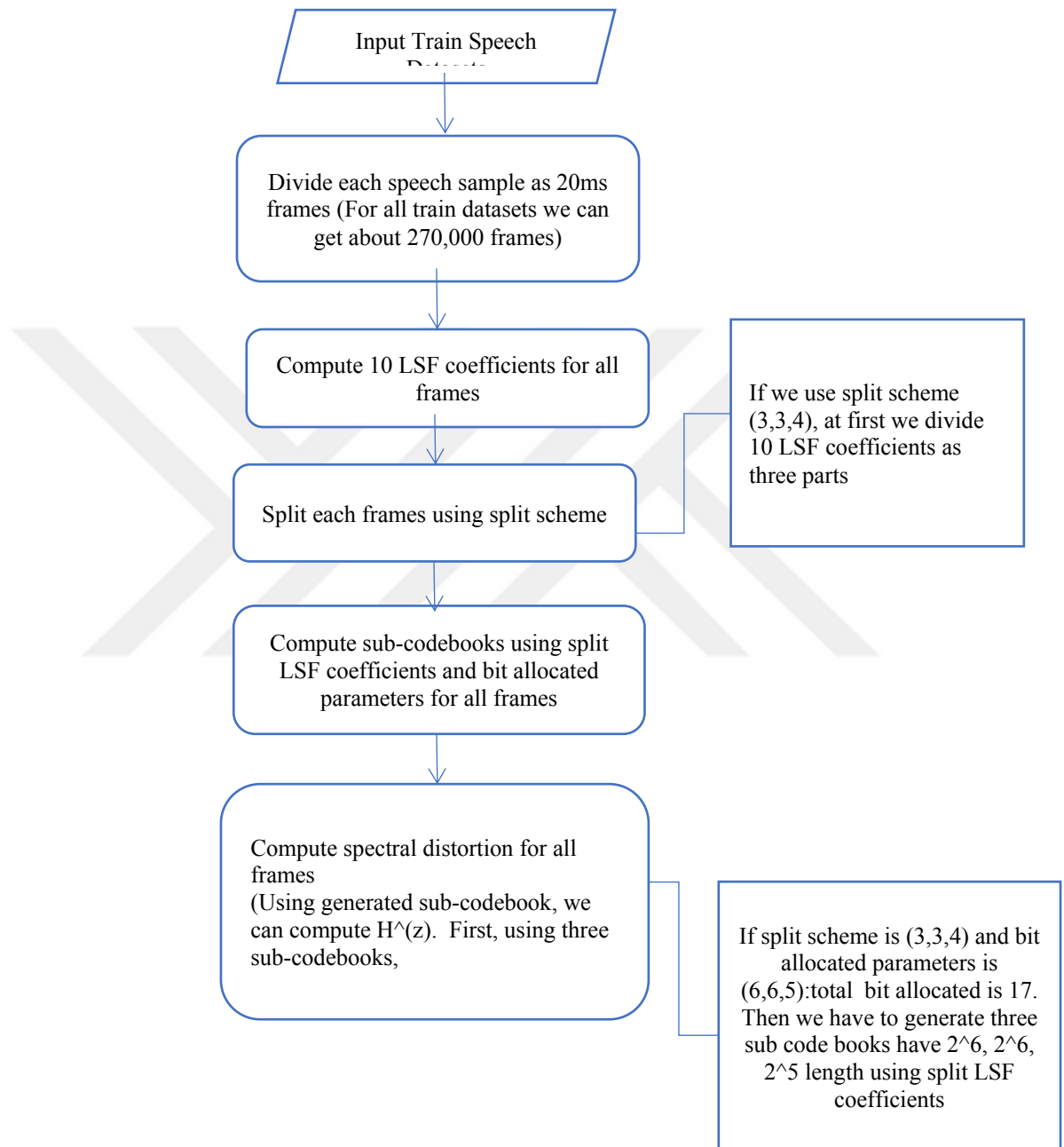


Figure 15. Flow chart for SVQ

6.2.1 Performance analysis for SVQ

In this study, different codebooks were designed using K-mean algorithm. The multistage vector quantization and split vector quantization methods are used to generate the codebooks of bit rates from 14 to 24 bits/frames. The SD is calculated using clean and noisy speech, the computation complexity and storage requirement have been calculated according to the equation given in (6.4),(6.6). In split vector quantization we used two and three splitting's. The results in Table 2 indicate that the best performance was obtained in (4,4,2) split codebooks when we compared to (3,3,4) and (4,3,3). According to these results, we can say that when we use large codebooks for the first and second split, quantization accuracy increases according to other equal size codebooks. Evaluation of the 3-split quantizers, relatively large codebooks can be used for the first and the second split. These codebooks should preferably be of equal sizes or alternatively one extra bit may be allocated for the second split. However, when the bit rate increases the largest allowed codebook size becomes as limiting factor and a few bits are wasted as they are forced to be allocated for the least significant third split. In our simulations, this limit was reached at the bit rate of 24, as the codebooks of the first two splits could not be enlarged and thus more than three bits had to be allocated for the last split. At higher bit rates, the best performance was obtained with the (4,4,2) splitting, where the largest codebook is required for the middle split and the codebooks for the first as well as for the third split can be of equal sizes or one extra bit can be allocated for the first split.

Table 1. Two split (SVQ) using clean speech

Bits/frame	SD (dB) for (6,4)	SD (dB) for (5,5)	SD (dB) for (4,6)
14	2.81	2.75	2.90
16	2.49	2.51	2.53
18	2.22	2.20	2.24
20	2.11	2.04	2.03

Table 2. Three split (SVQ) using clean speech

Bits/frame	SD (dB) for (3,3,4)	SD (dB) for (4,4,2)	SD (dB) for (4,3,3)
14	2.42	2.30	2.33
16	2.28	2.14	2.18
18	2.07	1.93	1.98
20	1.85	1.74	1.77
22	1.72	1.52	1.68
24	1.47	1.34	1.37

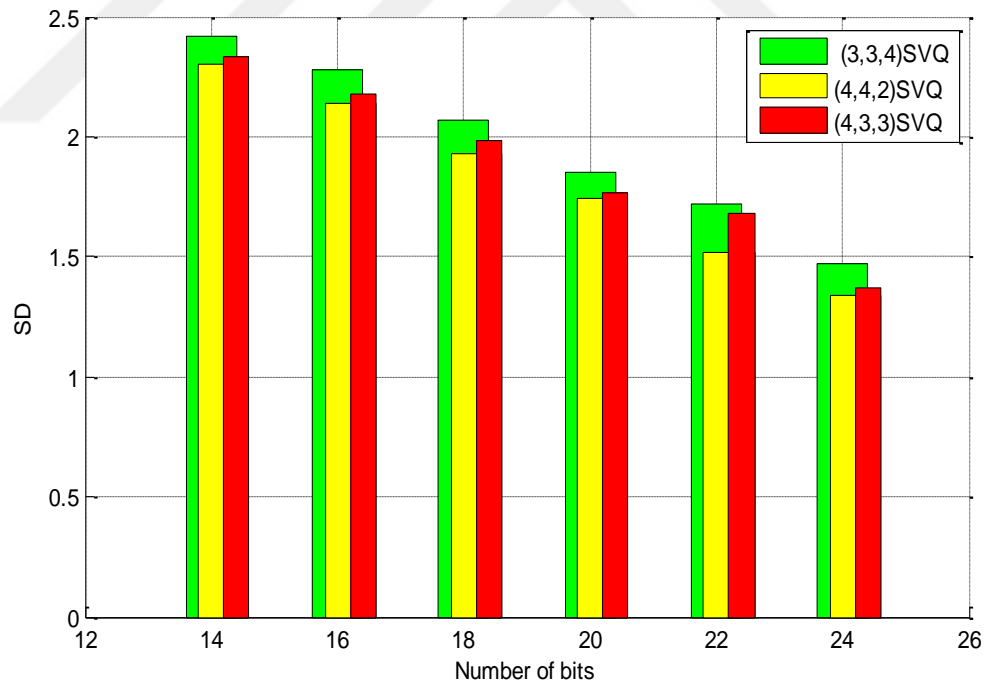


Figure 16. SVQ for clean speech

Table 3. Three split (SVQ) using noisy speech

Bits/frame	SD (dB)for (3,3,4)	SD (dB)for (4,3,3)	SD (dB)for (4,4,2)
14	4.22	4.64	3.92
16	3.94	4.55	3.81
18	3.76	3.91	3.53
20	3.38	3.22	3.30
22	3.06	2.98	2.88
24	2.91	2.73	2.68

6.3 Multistage vector quantization (MSVQ) Codebook design

In Multi-Stage Vector Quantizer (MSVQ), the input vector is quantized as a sum of vectors from a number of codebooks. Each of these codebooks can therefore be of relatively small size, making the storage requirements reasonable.

For design and testing of MSVQ codebooks, we have used 3-stage and 4-stage MSVQ codebooks for clean and noisy input speeches and compared SD result of each codebooks. The MSVQ codebooks in this study are designed using the M-L search, in which the M-best vector combinations are searched at each of the L stages, achieves performance close to that of the optimal search with a relatively low complexity. We used M=8 search depth for training and testing. In our algorithms, a large number of three or four stage codebooks with different numbers of bits and different bit allocations were trained and evaluated, [94-96]. The largest size of an individual codebook was selected as 8 bits (256 code vectors). The following observations were made based on our simulation results. When we compared 3-stage and 4-stage codebook SD results, it is seen that spectral distortion increases when stage numbers are increased for each bit/frame numbers. From these results, it can be said that, the number of stages should be kept as low as possible to minimize the spectral distortion value. As one can expected, the spectral distortion increased when we used the noisy input speech data for testing the codebooks as can be seen from Table 3.

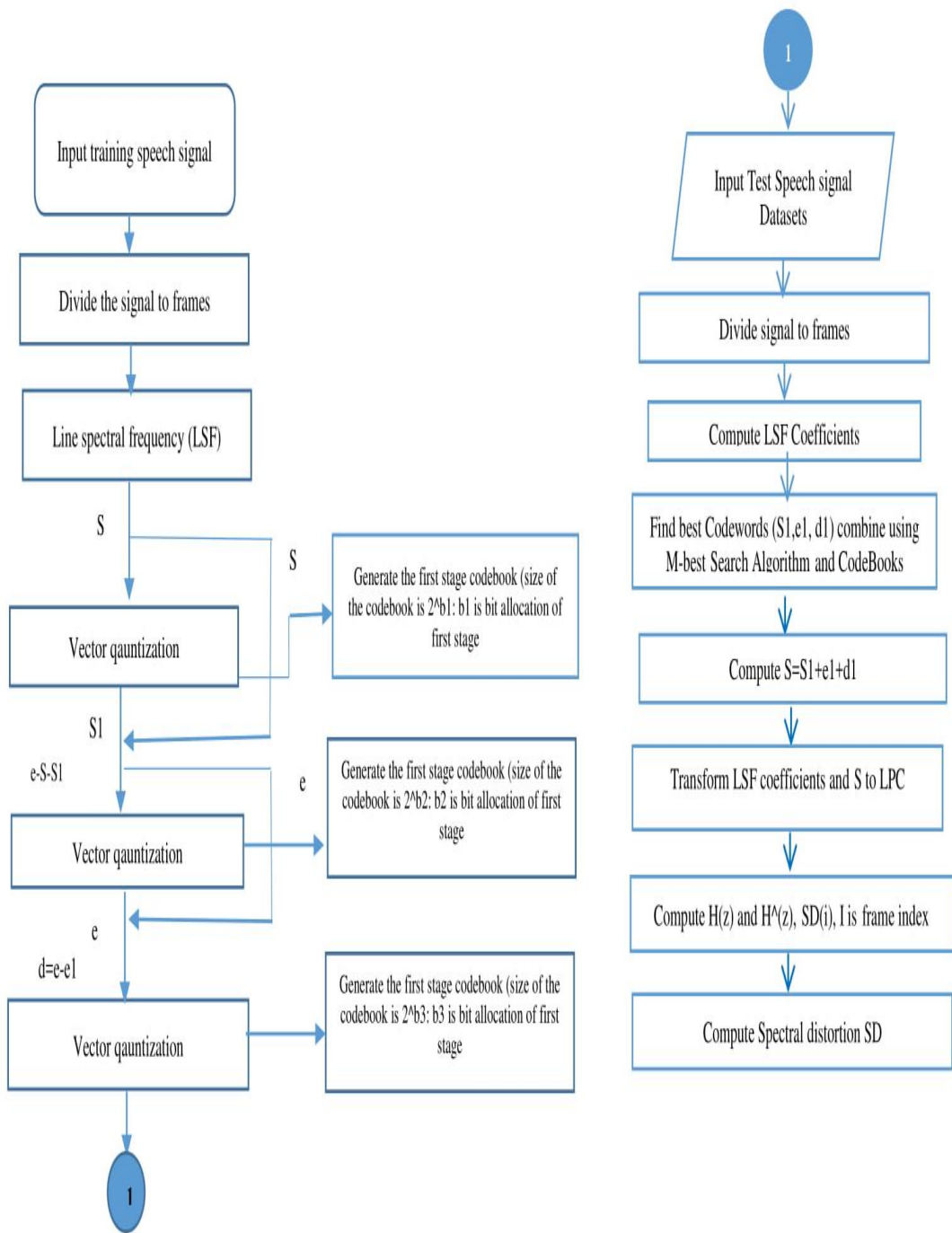


Figure 17. Flow chart for MSVQ

6.3.1 Performance Analysis for MSVQ

In multistage vector quantization the basic concept of vector quantization as applied to speech compression is schematically depicted in. A training speech sequence is 1st used to generate the codebook. Speech signal is segmented (windowed) into successive short frames and each frame of speech is represented by a vector. Tables 4,5 shows the spectral distortion (dB), at various bit rates for a 3-stage multistage vector quantizer and 4-stage multistage vector quantizer, as we can see from the tables that with 3-stage gives better performance than in 4-stages in terms of SD.

As we can see, from the table of MSVQ when we increase the number of stages, SD gets higher. Therefore, the best stage to have good SD is 3-stage, when we compare 3-stage for MSVQ and 3-split SVQ as we can see from figures and tables MSVQ gave better performance than SVQ using the same data for clean and noisy speech.

Table 4. Three and Four stages (MSVQ) using clean speech

Bits/frame	SD for 3-stage	Bits/frame	SD for 4-stage
14	1.94	14	2.09
16	1.80	16	1.95
18	1.61	18	1.82
20	1.43	20	1.62
22	1.31	22	1.44
24	1.19	24	1.26

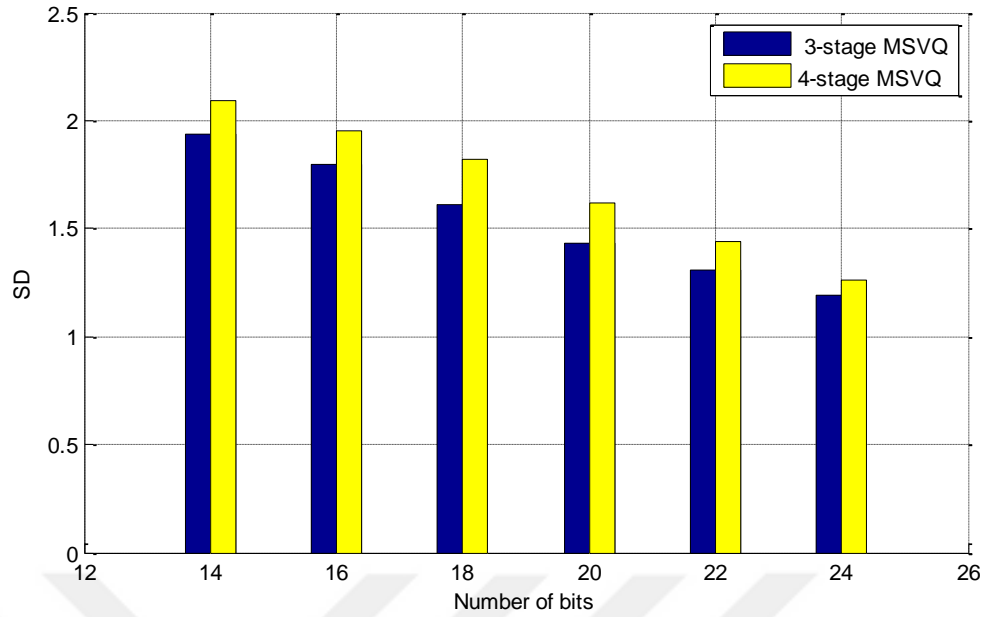


Figure 18. SD for MSVQ

Table 5. 3 and 4 stages (MSVQ) using noisy speech

Number of bits	SD for 3-stage	Number of bits	SD for 4-stage
14	3.25	14	3.87
16	2.92	16	3.70
18	2.83	18	3.45
20	2.40	20	3.28
22	2.17	22	2.91
24	2.01	24	2.83

6.4 Distortion, Complexity and memory measurement for the codebooks

To evaluate the performance of the MSVQ and SVQ codebooks, the spectral distortion (SD) was measured. Spectral distortion is one of the most frequently used objective measure technique for evaluating the performance of LSF quantizers. It is defined in dB as :

$$A(z) = 1 + \sum_{k=1}^p a_k z^{-k}, \quad \hat{A}(z) = 1 + \sum_{k=1}^p \hat{a}_k z^{-k} \quad (6.1)$$

$$d_{SD}(A(z), \hat{A}(z)) \approx \sqrt{\frac{1}{n_1 - n_0} \sum_{n=n_0}^{n_1-1} \left[10 \log_{10} \left(\frac{|\hat{A}(e^{j2\pi n/N})|^2}{|A(e^{j2\pi n/N})|^2} \right) \right]^2} \quad (6.2)$$

$$SD_{avg} = \sqrt{\frac{1}{T} \sum_{i=1}^T d_{SD}^2(e^{j2\pi i/N})} \quad dB^2 \quad (6.3)$$

Where $A(z)$ and $\hat{A}(z)$ denote the original and quantized p th order linear predictors while n_1 and n_0 correspond to lower (100Hz) and upper (3800Hz) frequencies respectively. $A(z)$ is the optimal p th order linear predictor and $\hat{A}(z)$ is the predictor with quantized coefficients. $N = 256$ -point FFT is used. The SD value is calculated over the bandwidth of 0-4kHz due to the 8kHz sampled input speech data.

6.4.1 Complexity

The Split VQ Complexity

$$VQ(SVQ) = \sum_{i=1}^{Sp} 4n2^b - 1 \quad (6.4)$$

The Multistage VQ Complexity

$$VQ(MSVQ) = \sum_{i=1}^m 4n2^b - 1 \quad (6.5)$$

Where b is the number of bits, n is the dimension of the vectors

6.4.2 Memory requirements

Split VQ is given

$$\text{Memory} = n2^b \quad (6.6)$$

Multistage VQ is given

$$\text{Memory} = \sum_{j=1}^p n^j 2^{bj} \quad (6.7)$$

6.5 Performance comparison of MSVQ and SVQ codebook results

The performance of two popular quantization methods, SVQ and MSVQ, was evaluated in LSF quantization. To achieve the best performance with MSVQ, it was found that the number of stages should generally be kept at a minimum provided that maximum size codebooks are used for each (but the last) stage. The evaluation of SVQ with 22 different splitting schemes indicated that none of the splitting schemes outperformed the others at all bit rates. The bit allocation was more complicated with SVQ than with MSVQ because the optimal allocation depends on both the sub-vector lengths and the frequency bands that the splits cover. The comparison between the quantization methods indicated that MSVQ clearly outperforms SVQ it can be seen from Figure 19. By using MSVQ instead of SVQ, at least 2–3 bits can be saved without any quality degradations. From Figure 20 comparison between MSVQ and SVQ in terms of complexity, we can see that SVQ give less complexity than MSVQ and Figure 21 shows the memory requirement is less in SVQ.

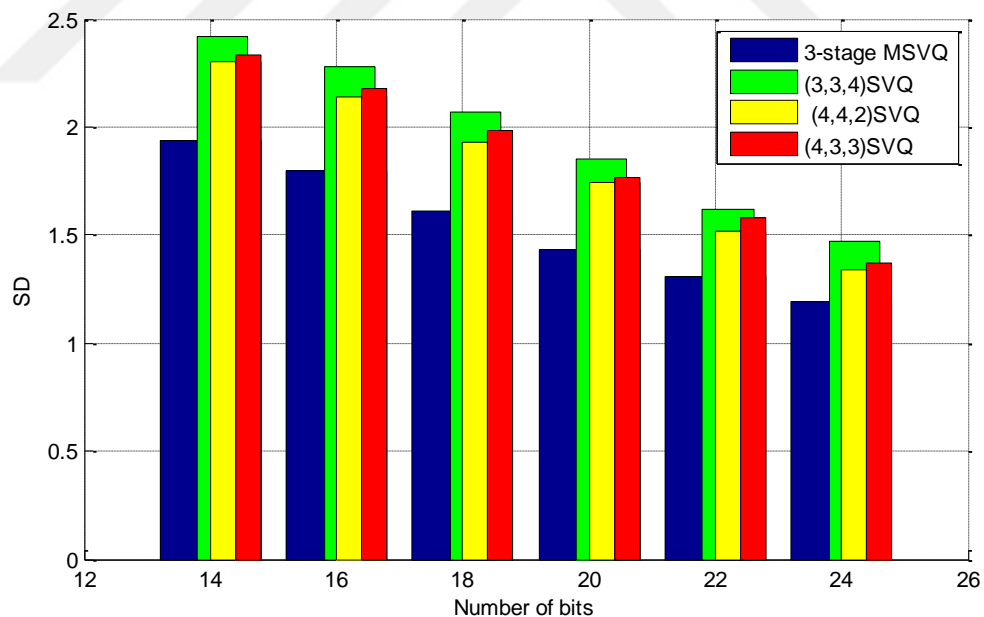


Figure 19. 3-stage MSVQ and 3-split SVQ for clean speech

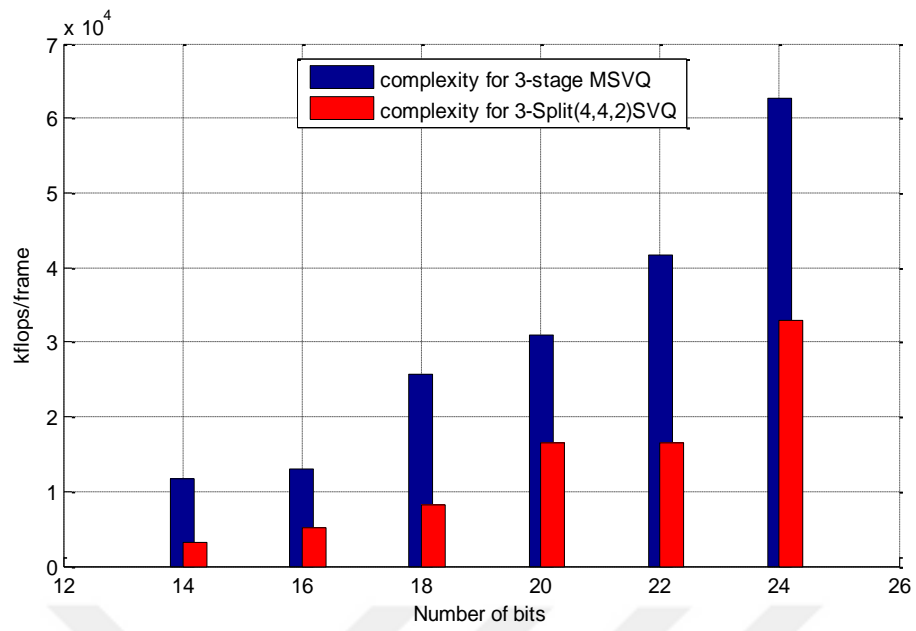


Figure 20. Complexity in MSVQ and SVQ

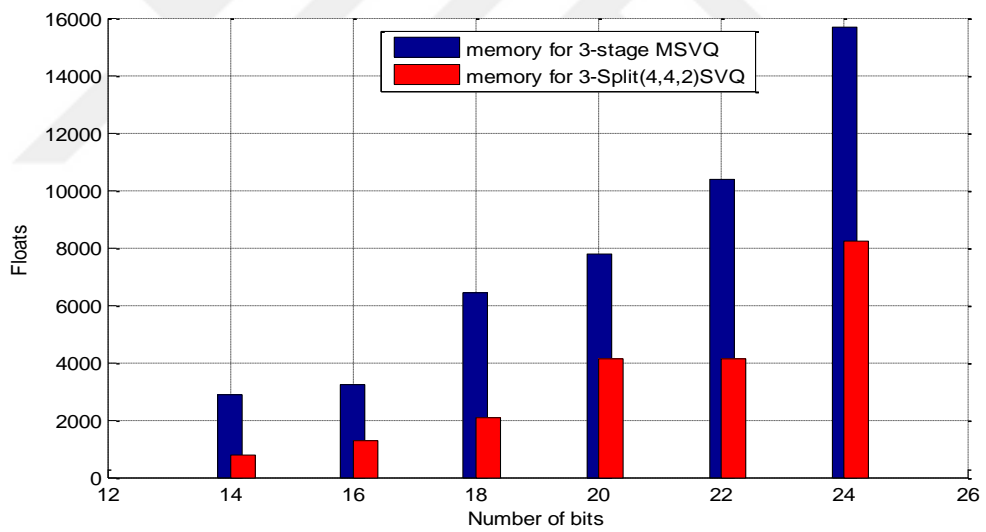


Figure 21. Memory requirement in MSVQ and SVQ

CHAPTER 7

RESIDUAL LSF QUANTIZATION METHODS

7.1 Introduction

VQ is a data reduction method, which means that in speech coding, quantization is required to reduce the number of bits used to represent a sample of speech signal. When less number of bits is used to represent a sample the bit-rate, complexity and memory requirement gets reduced. The challenge in VQ is to map an N-dimensional real-valued vector $x = [x_1, x_2, \dots, x_N]$ to another real-valued vector $y_i = [y_{i1}, y_{i2}, \dots, y_{iN}]$. Typically, all possible values of y_i are composed into a reconstruction codebook $Y = \{y_1, y_2, \dots, y_L\}$. Hence, if L is the size of the codebook, it is called an L-level codebook or L-level quantizer.

Vector quantization (VQ) is an important application of distance measures as it can reduce the dimension of an input data upon encoding each input vector as a single number. In a VQ algorithm, a set of training data is taken, and centroids are found by using a clustering algorithm. Among these clustering methods is the k-means clustering algorithm, which begins by choosing k vectors as the starting central points; then, it assigns each training input vector to the closest of these centers to gradually work out new central points by taking the means of each cluster. These three steps are repeated until the central point's cease to change any further or significantly. In speech coding, quantization is required to reduce the number of bits used to represent a sample of speech signal. When fewer bits are used to represent a sample, the bit-rate, complexity and memory requirement are reduced as well. Thus, a compromise must be made between the reduction in bit-rate and the quality of the speech signal. The parameters used in the analysis and synthesis of these signals are the LPC coefficients. In speech coding, quantization is not performed directly on the LPC coefficients, but carried out by transforming the LPC coefficients to other forms to ensure filter stability

after quantization. The alternative to LPC coefficients is the use of line spectral frequency (LSF) parameters for filter stability after quantization. [103-107]

Speech coding refers to the process of reducing the bit rate of digital speech representations for transmission or storage while maintaining a speech quality that is acceptable for the application. Most of the speech coders reported in the literature are based on linear prediction (LP) analysis [1]. For LP-based vocoders, the bit rate reduction is strongly tied to efficient quantization of the LPC filter coefficients $\{a_j\}$. The Line Spectral Frequencies (LSF), an equivalent representation of $\{a_j\}$ and more suitable for quantization and interpolation, can alternatively be used for the same purpose. In this sense, the Multi-Stage Vector Quantization (MSVQ) of LSF parameters presented in [104] has an efficient quantization performance at 22-24 bits per 20ms frames. Furthermore, such a multi-stage structure has more flexibility than a single stage VQ in terms of search complexity, codebook storage and channel error protection. Very low-rate speech communication systems require efficient fixed-rate and low delay coding methods which operate at lower bit rates.

The residual multistage vector quantization (RMSVQ) is a combination of two product code vector quantization techniques, namely the Residual vector quantization technique and the multistage vector quantization technique. In our study, the RMSVQ is implemented using an LSF coefficient vector split into stages, each of which employs an M-L search for quantization according to a multistage structure. The residual LSF parameters of a current frame are predicted from the quantized LSF parameters of the previous frames and, then, the residual LSF vectors are coded with an MSVQ codebook. RMSVQ offers satisfactory performance and reaches the transparency required for speech, hence its employment in the present study to improve the spectral distortion. For low bit-rate linear predictive speech coding, it is best to quantize the LPC parameters using as few bits as possible with transparent quantization quality. To reduce the costs associated with applying vector quantization (VQ), many structured schemes have been proposed among them the split VQ (SVQ) and the multistage VQ (MSVQ) schemes that offer substantial saving in both memory and computational cost. In our work we present to analyze these two methods, both of which employ an M-L search algorithm to significantly reduce the computational complexity and produce identical even improved quality with sequential search in

terms of spectral distortion and outliers. A comparison is made of four different VQ methods, namely (SVQ, MSVQ, RSVQ, and RMSVQ) all evaluated in terms of SD complexity and memory requirement. Simulation results show that a 3-stage split residual vector quantization can achieve a transparent quantization quality at 24 bit/frame. And MSVQ gives better performance in SD than SVQ in terms of SD, and RMSVQ gave the best SD of all methods specially when compared to RSVQ.

7.2 Design of Residual MSVQ and Residual SVQ codebooks

Speech signal VQs require codebook generation which, in this study, are designed using an iterative algorithm called the Linde, Buzo, and Gray (LBG) algorithm [54] as seen in chapter 4. The input to the LBG algorithm is a training vectors clustered into a set of codebook vectors. The speech signals used to obtain training vectors must be free of background noise, and they are ideally recorded in soundproof booths, computer rooms, and open environments. In this work, the speech signals are taken from the TIMIT database. The codebook generation using the LBG algorithm requires the formation of an initial codebook, which is the centroid or means obtained from the training sequence. This centroid is, then, split into two centroids or codewords using the splitting method. In turn, the iterative LBG algorithm splits these two codewords into four, four into eight, and the process continues till the required number of codewords in the codebook are obtained [108-110].

7.3 Residual Multistage vector quantization

Multistage Vector Quantization (MSVQ) is an evolution of the basic VQ technique, otherwise regarded as multistep, residual or cascaded vector quantization. It is a cascaded VQ encoder where the output of the VQ of a stage is the input of the next. It preserves the features of the VQ technique while reducing computational complexity and memory requirements while improving the quality. In MSVQ, each stage has its own codebook. The codebook of the first stage is created using the training sequence as the input, and the codebook of each remaining stage is created using the quantization error of its previous stage. As for VQ, we used the LBG algorithm to create the codebooks. Let y be the signal to be quantized and assume the number of stages as 3 ($st = 3$) which illustrates the MSVQ encoding and decoding level. Note that, here, the input of the VQ of the first stage is y , the input of the VQ of the second stage is the

quantization error of the previous stage $e_1 = y - \hat{y}_1$, and the input of the VQ of the third stage is the quantization error of stage two $e_2 = e_1 - \hat{e}_1$. The MSVQ encoder provides three indexes as the observation vector (y), the quantization error of the first stage (e_1), and the quantization error of the second stage (e_2). The MSVQ decoder uses the codebook of each stage to find the corresponding codewords of the indexes and, then, provide the quantized vectors of the three stages as \hat{y}_1 , \hat{e}_1 , and \hat{e}_2 . Accordingly, the quantized observation vector is $\hat{y} = \hat{y}_1 + \hat{e}_1 + \hat{e}_2$. In this study, we designed the residual MSVQ (RMSVQ) algorithm, with the related flowchart appearing in Figure 22.

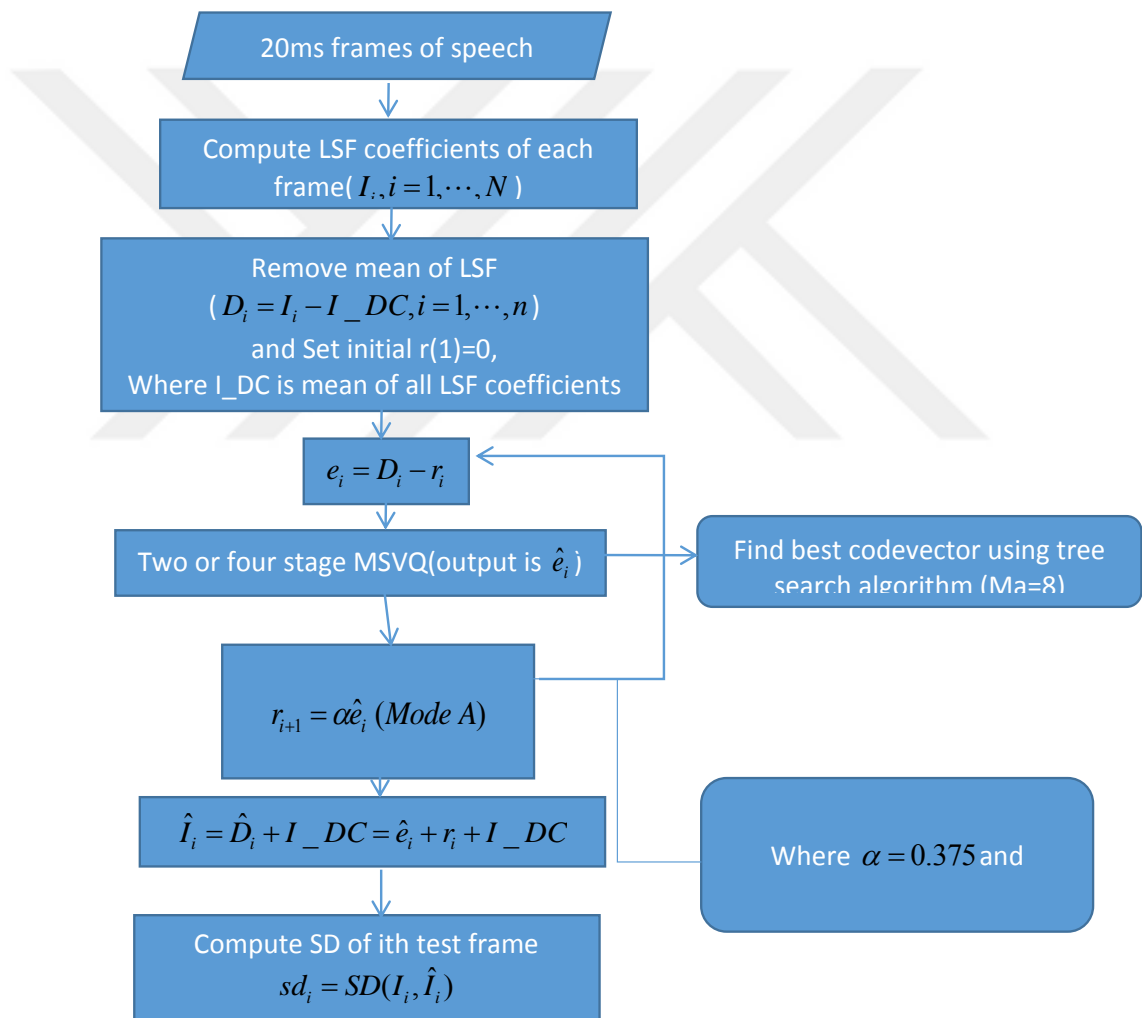


Figure 22. Flow chart for RMSVQ

As to the RMSVQ, the following details describe the approach:

- ❑ RMSVQ is a hybrid of two product code vector quantization techniques; namely, the residual vector quantization technique (RVQ) and the multistage vector quantization (MSVQ) technique.
- ❑ The RMSVQ implemented employs LSF coefficients. Here, the LSF coefficient vector is split into stages, where M-L search is used according to a multistage structure.
- ❑ The residual LSF parameters of the current frame are predicted from the quantized LSF parameters of the previous ones and, later, residual LSF vectors are coded with an MSVQ codebook.

In the RMSVQ method, the residual LSF parameters of the current frame are predicted from the quantized LSF parameters of the previous frames using interframe correlation feature of spectrum parameters [107-110] and then residual LSF vectors are coded with a MSVQ codebook. Firstly, the LSF parameter vector is obtained by transforming a 10th order LPC parameter vector. Next, the long-term average LSF vector (obtained by averaging the LSF vectors in the training set) I_{DC} is subtracted from the LSF vector belonging to the i th frame I_i to obtain a differential LSF vector d_i given by :

$$d_i = I_i - I_{DC} \quad (7.1)$$

Where $i=1,2,\dots$ and $r(0) = 0$. The quantized residual vector ($e^{(i)}$) is found by quantizing $e^{(i)}$ with a VQ codebook. Depending on how (i) is computed, various prediction schemes can be proposed. Scalar quantity α with a low value of 0.375 is used as the correlation coefficient for backward prediction of the LSF residual vector.

$$e_i = d_i - r_i \quad (7.2)$$

$$r_i = \alpha e^{(i)} \quad (7.3)$$

$$I_i = d_i + r_i \quad (7.4)$$

$$= e^{(i)} + r_i + I_{DC} \quad (7.5)$$

Table 6. SD for RMSVQ for 3 & 4 splits

Number of bits	SD for 3-stages	SD for 4-stages
14	1.84	2.06
16	1.71	1.87
18	1.53	1.68
20	1.33	1.44
22	1.21	1.34
24	1.08	1.22

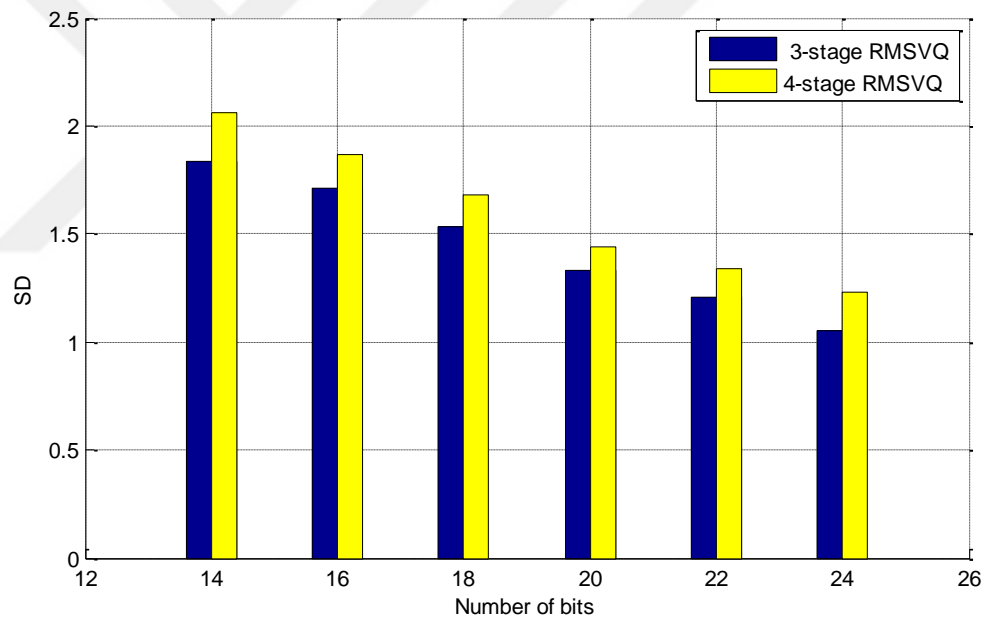


Figure 23. 3-stage and 4-stage RMSVQ

7.4. RESIDUAL SPLIT VECTOR QUANTIZATION

This technique is developed to improve the performance of split vector quantization. As stated earlier, Residual split vector quantization (RSVQ) is also a hybrid of two product code vector quantization techniques, namely the residual vector quantization technique and the split vector quantization technique. In RSVQ, the vector dimensions to be quantized are reduced by means of splitting, and the bits allocated to the quantizer are divided among the stages and splits of each stage. All sub-vectors in the quantizer are connected in cascade to form a multi-stage structure. Obviously, splitting the input vector into sub-vectors will lead to performance degradation because of the decline in the statistical dependence between the sub-vectors. However, if an analysis procedure is devised in such a way to allow for coupling the forward and backward prediction errors across successive stages, then the statistical dependence can be preserved during the quantization performance of the SVQ to approach to that of a single stage VQ.

- ❑ In Residual Split Vector Quantization, the training sequence used for codebook generation is split into vectors of smaller dimensions.
- ❑ Each split of the training sequence is used to generate separate sub codebooks.
- ❑ Computing LSF coefficient and removing the mean from the LSF coefficient.
- ❑ Each VQ stage operates on the residual vector of the previous stage similar to the residual VQ scheme.
- ❑ The use of a split vector quantizer makes the less availability of bits at each split of the vector quantizer

As a result, the complexity and memory requirements are greatly reduced, but the dependencies that exist across the dimensions (splits) of a vector is lost. Consequently, the spectral distortion is slightly increased.

Table 7. SD for RSVQ for 3-splits

Number of bits	SD for (3,3,4)	SD for (4,4,2)	SD for (4,3,3)
14	2.31	2.17	2.20
16	2.24	2.08	2.10
18	2.02	1.86	1.96
20	1.80	1.61	1.71
22	1.59	1.44	1.52
24	1.43	1.28	1.34

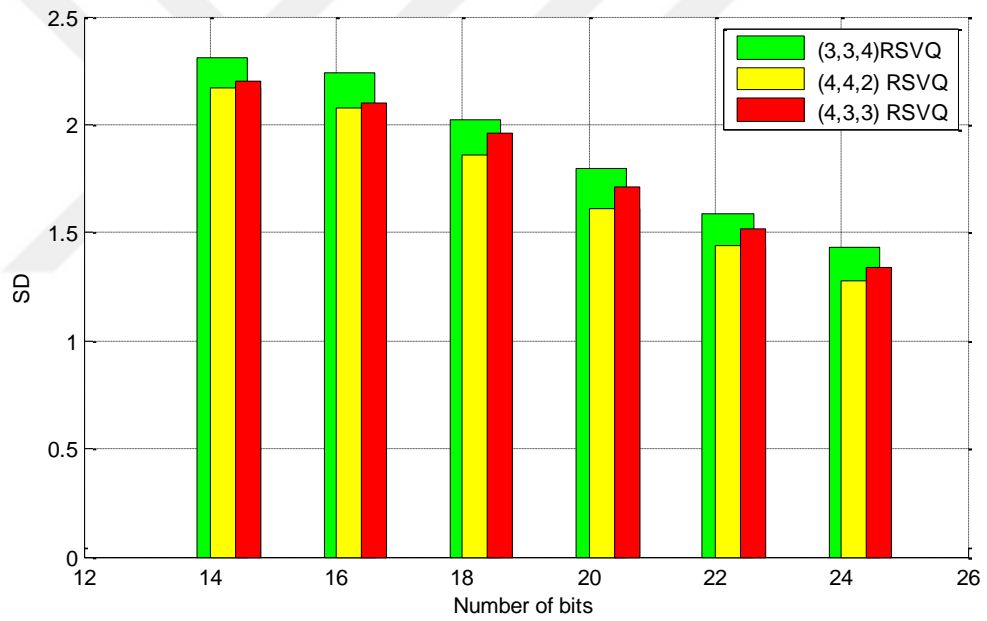


Figure 24. 3-split RSVQ

Table 8. SD for RSVQ for 4-splits

Number of bits	SD for (4,2,2,2)	SD for (3,3,3,1)	SD for (3,3,2,2)
14	2.643	2.457	2.363
16	2.475	2.339	2.182
18	2.092	2.190	2.026
20	1.937	1.930	1.801
22	1.734	1.735	1.587
24	1.663	1.648	1.515

Table 9. The SD for RMSVQ noisy speech

Number of bits	SD for 3-stage	SD for 4-stage
14	2.83	3.08
16	2.54	2.93
18	2.39	2.74
20	2.18	2.53
22	2.05	2.34
24	1.95	2.19

7.4 PERFORMANCE EVALUATION

The quality of the speech signal is an important parameter in speech coders, measured in terms of spectral distortion for an objective performance evaluation. The spectral distortion is measured between the LPC power spectrum of the quantized and unquantized speech signals. In this work, the experiment is carried out using the standard TIMIT speech database containing American and British English speeches sampled at 8 kHz. In order to design SVQ or MSVQ codebooks, the LSF vectors are generated from the TIMIT database. The sentences are uttered by both female and male speakers. The

frame length is about 30 ms with hamming window and overlap 10ms. A 10th-order LPC analysis based on an autocorrelation method is carried out for every 20 ms frame. The resulting coefficients of the 10th-order LP polynomial $A(z)$ are, then, converted into the LSF domain. The LSF input data is used to design the required codebooks. The input data used in the experiments consists of two separate databases, namely for training and testing. The former consisted of 270,000 LSF vectors for clean speeches, and the latter consisted of 108,300 LSF vectors for clean speech and 54,700 LSF vectors for noisy speech. After designing the codebooks, the spectral distortion (SD) values are calculated over the frequency band of 100-3800Hz for 8 kHz sampled speech [103].

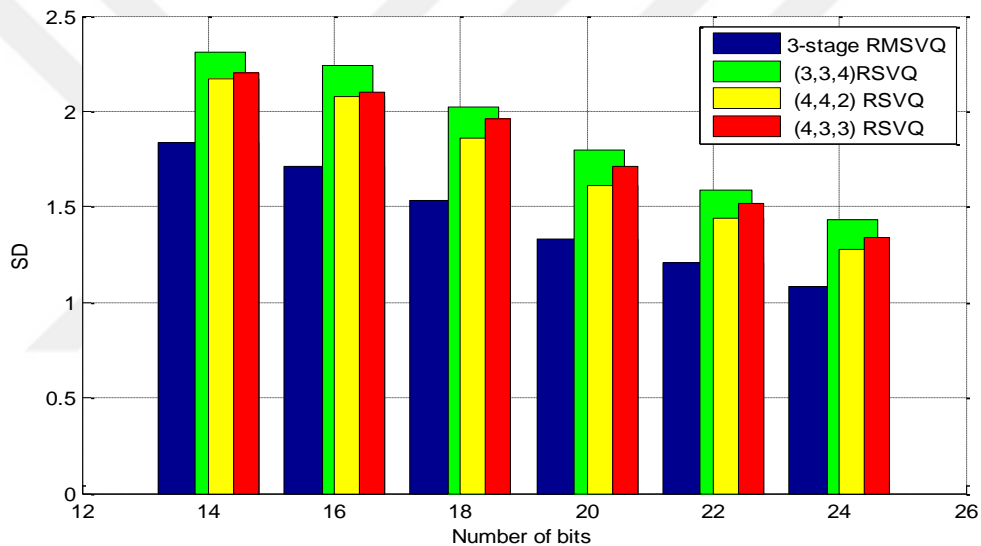


Figure 25. SD for 3-stage RMSVQ and RSVQ

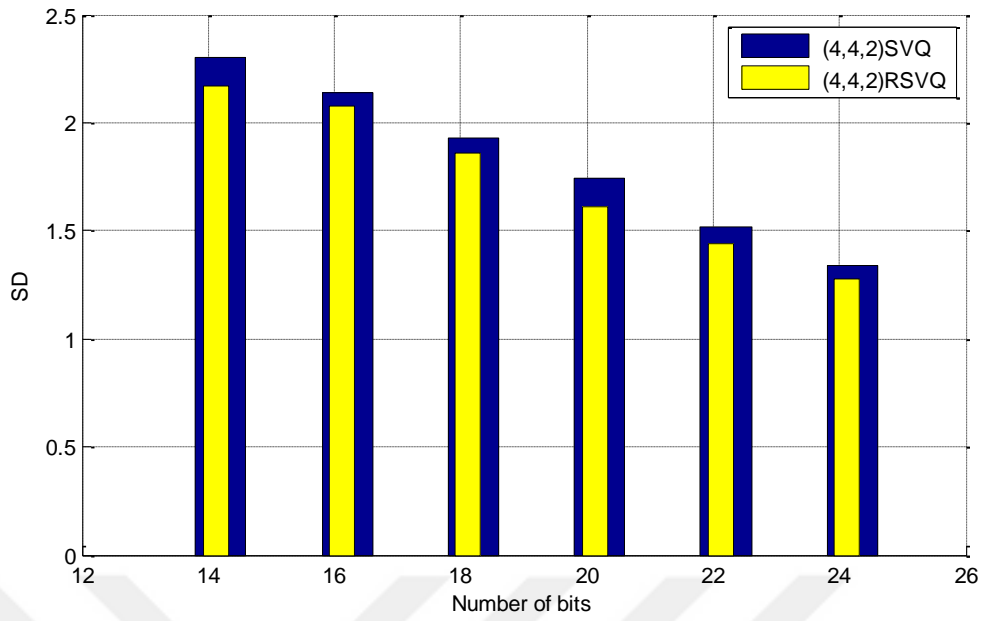


Figure 26. SD for 3- stages RSVQ and SVQ

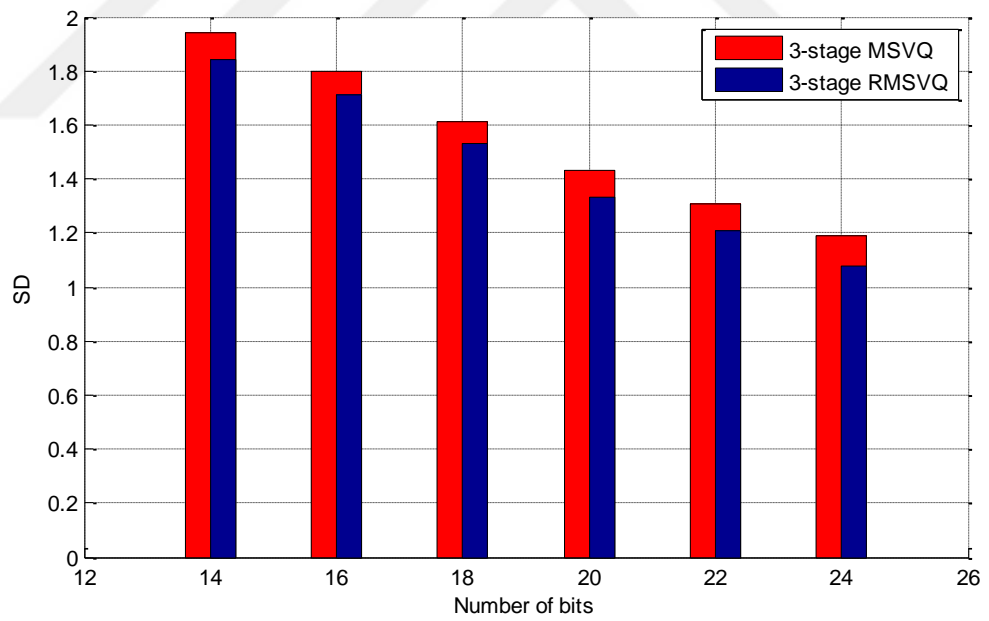


Figure 27. SD for 3- stages RMSVQ and MSVQ

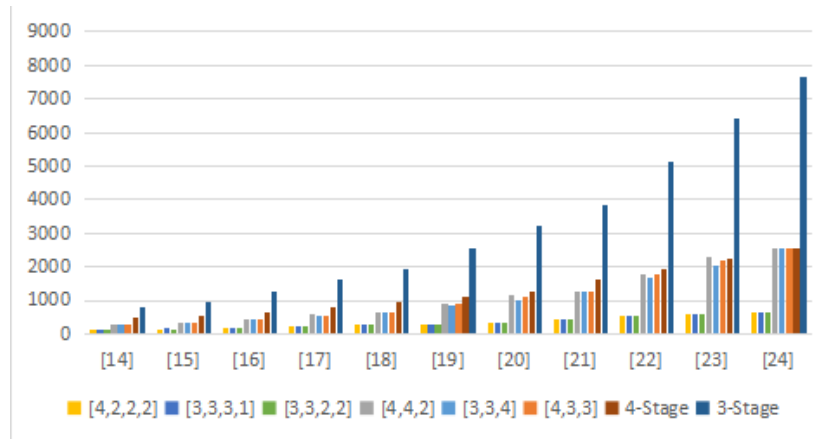


Figure 28. Memory requirement comparison for all stages in (MSVQ and SVQ)

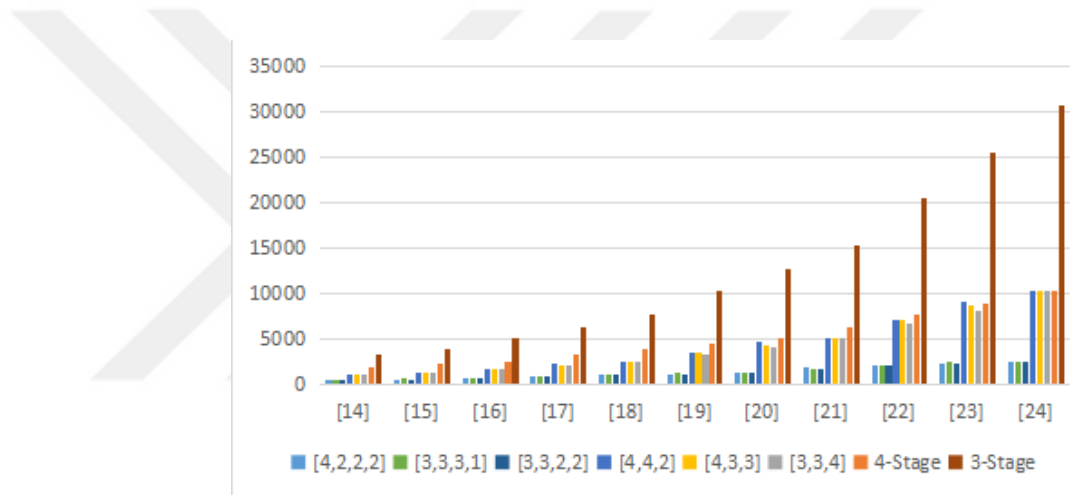


Figure 29. Complexity comparison for all stages in (MSVQ and SVQ)

7.5 Performance Analysis

As we can see from table 6,7 and 8 calculating the SD in all number of bits from 14 to 24 RMSVQ give very good performance in terms of SD when compared with RSVQ. And as we can see from previous chapter that MSVQ is better than SVQ in terms of SD and comparing previous results for (SVQ, MSVQ) with new methods (RSVQ, RMSVQ) from the table results and figures RMSVQ gives the best result when compared with all algorithms.

CHAPTER 8

DESIGN OF RMSVQ CODEBOOK WITH VOICED/UNVOICED EVALUATION

8.1 Introduction

When analyzing speech, the voiced-unvoiced determination is often required to elicit data related to speech signals. The present thesis, with this in mind, employs two different attributes to distinguish voiced segments from unvoiced ones; these are the zero crossing rate (ZCR) and the energy. Then, the outcomes are assessed by breaking the sample down into parts and completing ZCR and energy analyses for our purpose. Accordingly, ZCR values are low in case of the voiced segments and high for the unvoiced ones; the energy, on the other hand, is precisely the other way around, hence proving the efficiency of the approach adopted. As speech comprises numerous voiced and unvoiced segments, its analysis according to these regions offers us insight into the basic acoustic structure of processing applications among them, speech synthesis, enhancement, or recognition operations.

Lately, experts have made strides toward tackling speech classification in the format mentioned earlier [85]. Pattern recognition techniques along with statistical/non-statistical methods are now used to determine the voiced-unvoiced nature of speech parts under review. In this regard, Qi and Hunt carried out similar studies by applying non-parametric techniques based upon multi-layer feed forward networks. In that work, acoustical attributes and pattern recognition models helped to distinguish the parts as voiced/unvoiced. In the present study, a more basic and easier technique is adopted for the same purpose by benefitting from the ZCR and energy measurement.

8.2 Voiced/Unvoiced evaluation of speech signal

As stated earlier, ZCR plays a key role as classification device for voiced/unvoiced patterns – mostly applied as part of front-end operations in automatic speech recognition

processes. ZCR values in this way reflect the frequency at which the energy is focused within the signal band. Voiced speech is a product of vocal tract movements due to the periodic flow of air at the glottis, often characterized by reduced ZCR values [9]. On the other hand, unvoiced segments are a product of tightening vocal tracts to the point that the airflow becomes unsteady and generates noise with added ZCR. Separately, energy in the voiced segments is of increased values owing to the periodicity factor and, naturally, of reduced values when it comes to unvoiced segments.

8.2.1. Zero-Crossings Rate

Discrete-time signals comprise ZCR events on the condition of continuous samples occurring with various algebraic signs. This rate is a basic function of the frequency within the signal; that is, a measure of the number of times within a period when the entire signal scale travel through a value of zero, as depicted in Figure 31. Speech signals are of broadband range and, in this way, it is not quite easy to accurately estimate average ZCR values. Nonetheless, a general measurement related to the band features is possible by means of representations related to short-interval approximates [91].

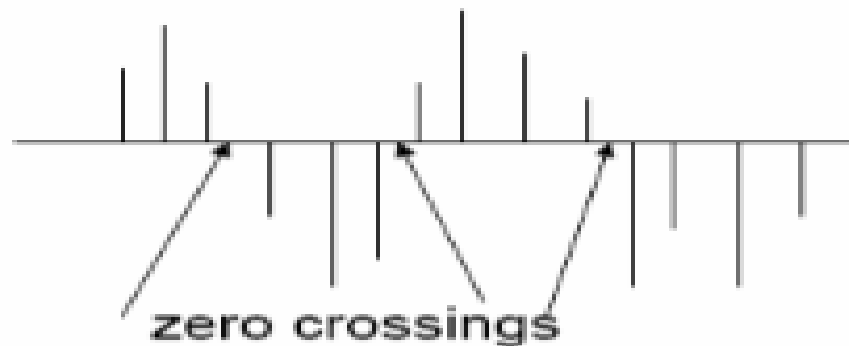


Figure 30. Definition of zero-crossings rate

$$Z_n = \sum_{m=-\infty}^{\infty} | \text{sgn}[x(m)] - \text{sgn}[x(m-n)] | w(n-m) \quad (8.1)$$

$$\text{sgn}[x(n)] = \begin{cases} 1, & x(n) \geq 0 \\ -1, & x(n) < 0 \end{cases} \quad (8.2)$$

$$-1, x(n) < 0$$

and

$$W(n) = \begin{cases} 1/2N & \text{for } 0 \leq n \leq N-1 \\ 0 & \text{for, otherwise} \end{cases} \quad (8.3)$$

According to the speech production model, the voiced segment energy appears lower than 3 kHz due to the band range drop by the glottal wave, though unvoiced segments' energy appears in frequencies of upper levels. Given that these frequencies also have added ZCRs, and low frequencies less ZCR, a clear and evident correlation is established between ZCR and energy distribution with frequency. One can only deduct, therefore, that a high ZCR means an unvoiced signal, and a reduced ZCR implies a voiced one [92].

8.2.2. Short-Time Energy

Speech signal scales change with time, often being lower in case of unvoiced segments compared to voiced ones as we can see from figure 31. The signal energy contains a representation that shows such scale changes. In detail, short-time energy can be described in the following way:

$$E_n = \sum_{m=-\infty}^{\infty} [x(m)w(n-m)]^2 \quad (8.4)$$

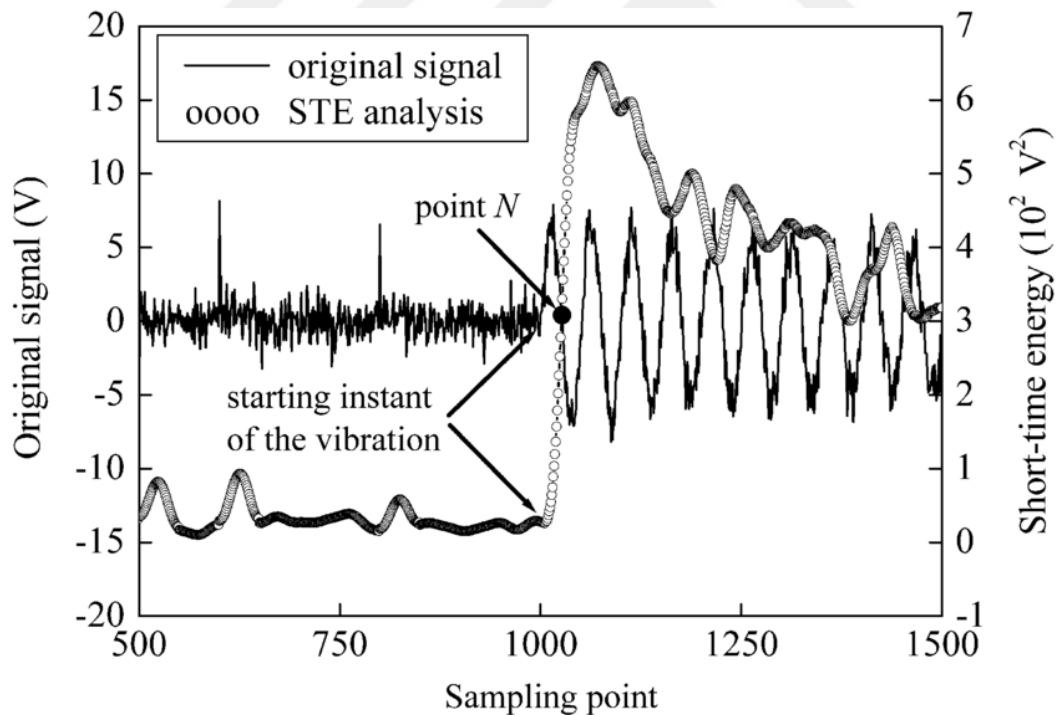


Figure 31. Definition of short time energy

8.3 Mode Classification

Here, different modes are assigned to each frame and described in general as spectrally stationary voiced mode (mode A) and unvoiced mode (mode B). This scheme is made for consistency within the different levels of clear and noisy speech. Larger speech datasets known as TIMIT files are applied for training the VUV_RMSVQ codebooks, and they contain utterances at different levels and noise values. The test speech data (exceeding 270,000 frames) is sent along the front-end mode classification scheme, and then the quantized LSF vectors are redesigned with the VUV_RMSVQ codebooks. The quantized and original LSF vectors are then compared based on averages and outlier percentages.

8.3.1 Algorithm design for the proposed method

The LSF parameter vector can be gained by transforming a 10th order LPC parameter vector. Then, the long-term average LSF vector as determined by averaging the LSF vectors in the training set, I_{DC} , is removed from the LSF vector of the i th frame I_i so as to get a differential LSF vector d_i as seen in (1):

given by (8.5):

$$d_i = I_i - I_{DC} \quad (8.5)$$

After separating the speech signal into voiced and unvoiced, we calculate the spectral distortion for voiced using diagonal matrix A as shown in (8.6).

$$A^{\wedge}[j][j] = \frac{\sum_{i=1}^N d_i[j] \cdot d_{i-1}^T[j]}{\sum_{i=1}^N d_{i-1}^2[j]} \quad (8.6)$$

Where N is the number of frames in the training set. The LSF residual vector for the i th frame e_i ; in the case of spectrally stationary mode A frames, is obtained as

$$e_i = d_i - A^{\wedge} \cdot d_{i-1} \quad (8.7)$$

For unvoiced (mode B) frames, adjacent frame LSF vectors are not well correlated. Scalar quantity α with a low value of 0.375 is used as the correlation coefficient for backward prediction of the LSF residual vector. In this case, the LSF residual vector is given by

$$e_i = d_i - \alpha \cdot d_{i-1} \quad (8.8)$$

Speech can be divided into numerous voiced and unvoiced regions. The classification of speech signal into voiced, unvoiced provides a preliminary acoustic segmentation for speech processing applications, such as speech synthesis, speech enhancement, and speech recognition.

In this research, residual LSF vectors are obtained by speech mode-based backward prediction along with a multi-stage VQ design. Next, the speech is grouped as voiced and unvoiced by training with two codebooks and also by testing with codebooks to estimate the SD for clear and noisy speech and outliers. As stated earlier, the two approaches to distinguish voiced segments from unvoiced ones are ZCR and energy analysis. The two related codebooks are formed at the training stage and another two in the testing period [29, 31]. Three algorithm are, then introduced with the aim to enhance the vector quantization performance based on the voiced/unvoiced classification.

8.3.2 Voiced/Unvoiced residual Multistage Vector Quantization Codebook Design

The VUV_RMSVQ codebooks are trained with the help of large speech datasets comprising different speakers. The parametric representation here is based on the residual LSF vector created in accordance to the speech mode following 1st order backward forecasting. The mode classification scheme, mode-based elicitation of the LSF residual vector, codebook format and search methods appear in the upcoming sub-sections.

The iterative sequential design method applied here trains the multi-stage VQ's in the same way as explained in chapter 6, comprising two stages:

- Initially, we develop a group of multi-stage codebooks sequentially, which means that the developed codebook employs a training set comprising those quantization error vectors of the preceding stage. Apparently, the 1st stage codebook will make use of the training set related to the LSF residual vectors. For all these trainings, the famous Lloyd algorithm will be employed.
- Next comes the iterative re-optimization of each stage to reduce the noise factor all across the stages. given an already existing set of multi-stage codebooks, each stage is improved based on the other stages; that is, the training set for each stage requires the quantization error between the input LSF residual

vector and a reconstruction vector comprising the lowest distortion codebook vectors belonging to all stages, excluding the one being re- optimized. The procedure is followed iteratively up to the point where a pre-determined convergence criterion can be achieved.

During training, all codebooks are designed based on voiced and unvoiced features, and later checked by means of a fixed stage different bit-rate. In case of the voiced speech mode A frames, a two-stage 12 bit VQ (64 vectors per stage) and a two-stage 14 bit VQ (128 vectors per stage) are formed with the help of a subset of the training data categorized as mode A (about 340,000 vectors). As to the remaining modes, a four-stage 22 bit VQ (64 vectors per stage for the first two stages, and 32 vectors per stage for the last two stages) and a four-stage 24 bit VQ (64 vectors per stage) are developed using all the dataset (about 1 hour).

8.3.3 Codebook Search and Quantization

During training and encoding, the multistage codebooks are investigated by applying an M-L tree search process defined in [117]. In this process, given the 1st stage codebook, M (M = 8 has been seen through tests to be satisfactory) codebook vectors obtaining the least distortion are chosen initially. Later, the M quantization error (from the 1st stage) vectors are calculated and the 2nd codebook is investigated with the help of the M error vectors, and M paths with the overall lowest distortion are chosen. The process is repeated for all stages of the codebook. After we find the M paths for all the stages, the best one is selected by reducing the distortion measure between the input LSF residual vector and the overall quantized vector which, for every path, is the total of the code vectors for all stages of the codebook. Lastly, the related indices for selected code vectors from each stage are sent to the speech decoder.

8.4 Residual multistage VQ for (Voiced/Unvoiced) speech using fixed size of codebooks

In this method we have used different size of codebooks and bit rate to see the effect on the SD after separating the speech to voiced and unvoiced, the flowchart below shows the steps of our method for VUV_RMSVQ.

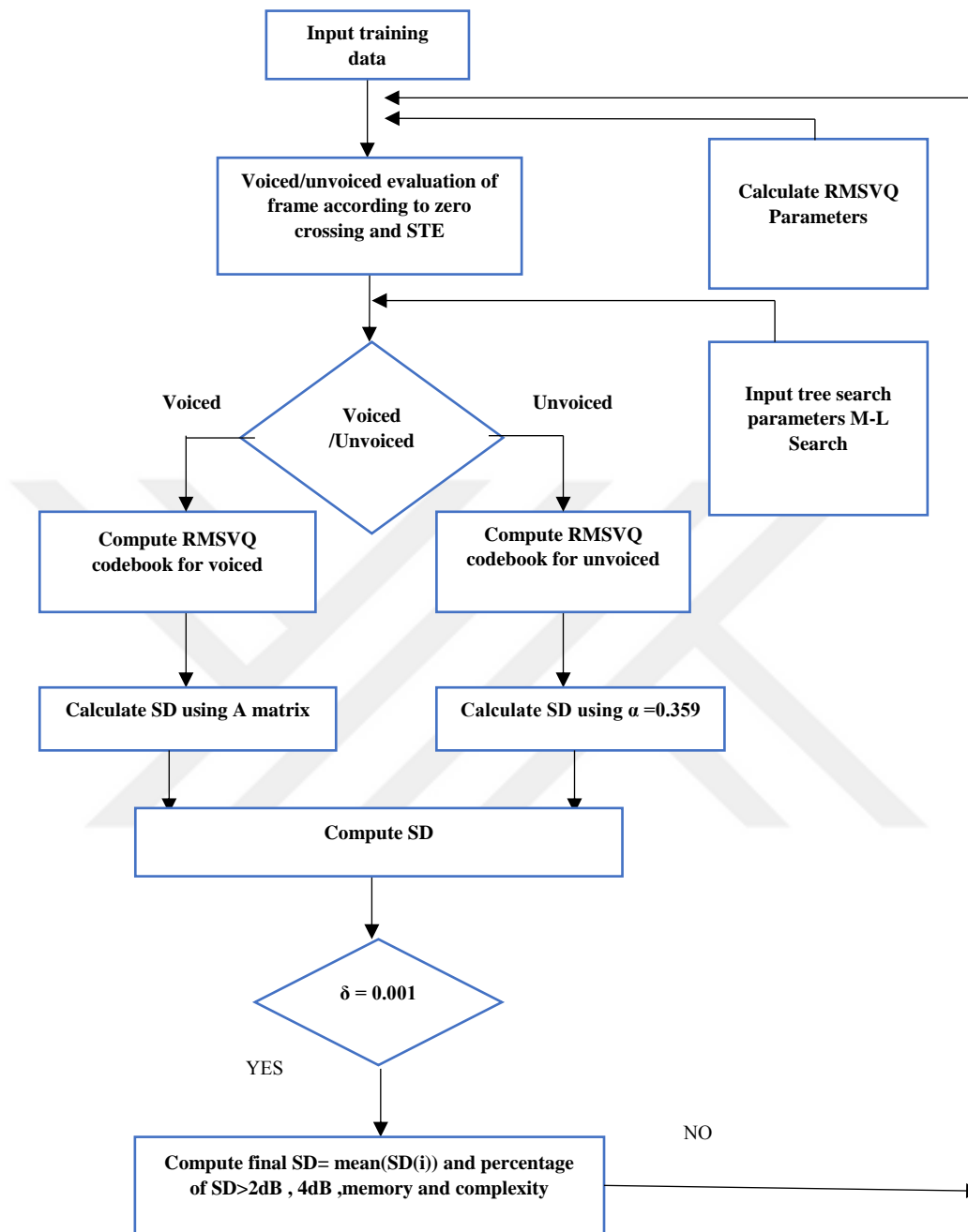


Figure 32. Flow chart for VUV_RMSVQ

Table 10. SD for VUV_RMSVQ

Number of bites	SD (3-stage)	SD (4-stage)	SD > 0.2	SD > 0.4
14	1.80	1.94	43.682	1.00
16	1.63	1.78	33.45	0.69
18	1.47	1.62	22.20	0.65
20	1.30	1.41	16.29	0.27
22	1.18	1.28	11.74	0.17
24	1.08	1.14	8.12	0.00

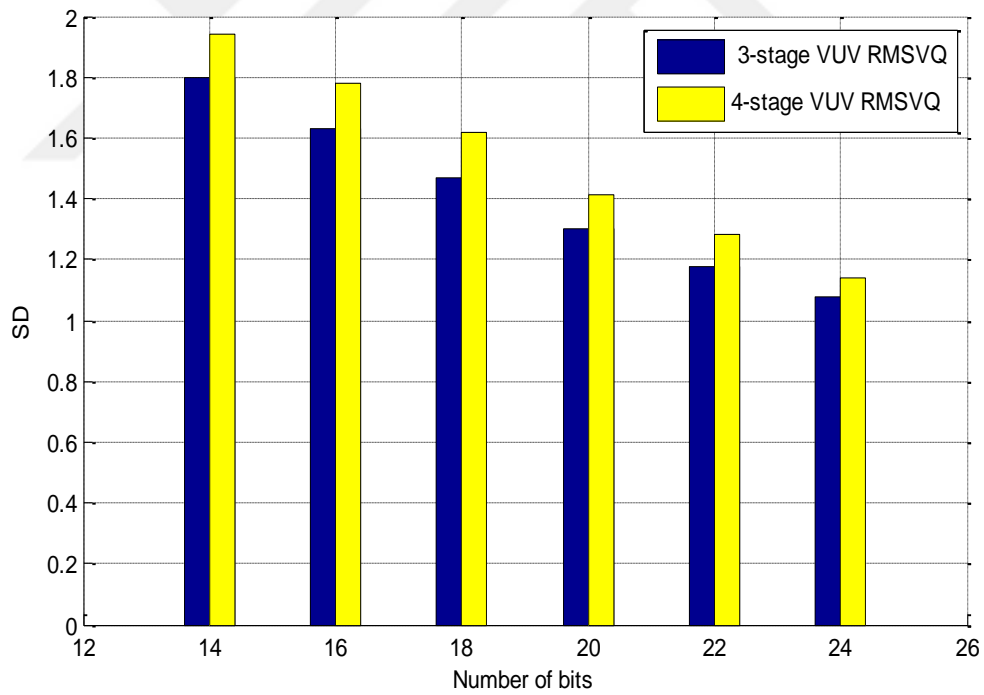


Figure 33. (3, 4 stages) for VUV_RMSVQ

8.5 Residual multistage vector quantization using different size of codebooks

The spectrally stationary Mode A (Voiced) frames will be used to train a 2-stage MSVQ and the Mode B (Unvoiced) frames will be used to train a 4-stage MSVQ. Two RMSVQ codebook were designed at the training stage for both voiced /unvoiced, and at the test stage. When separating the speech to voiced /unvoiced about 80% is for voiced and 20%for unvoiced. Figure 34 shows the design of our method.

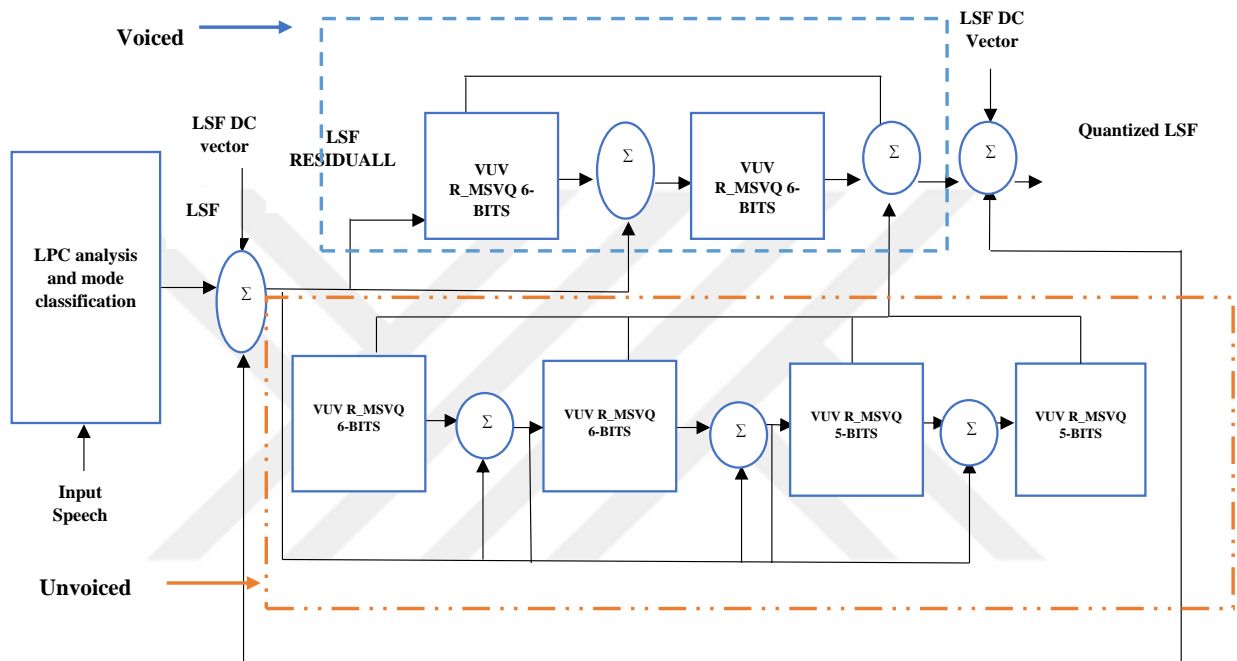


Figure 34. Block diagram for VUV_RMSVQ

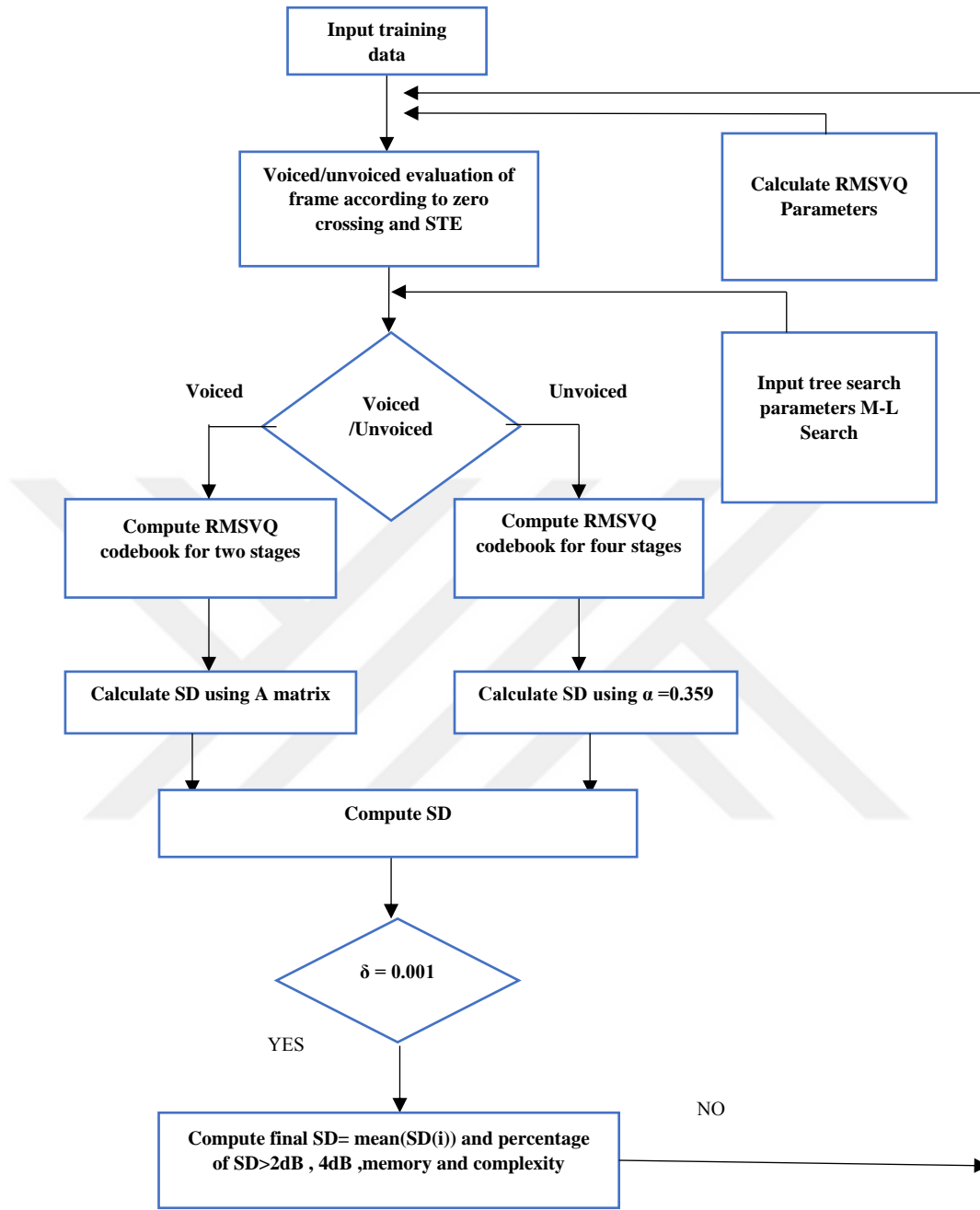


Figure 35. Flow chart for VUV_RMSVQ (2-4 stage)

Table11. SD for VUV_RMSVQ (2,4 stags)

Number of bits	Average bit rate	SD	SD > 0.2	SD > 0.4
9/12	9	2.3	25.3	1.00
11/14	11	2.1	18.06	0.64
12/16	12	1.92	13.5	0.44
14/18	14	1.73	10.0	0.30
15/20	16	1.56	7.2	0.2
17/22	18	1.42	6.1	0.17
18/24	19	1.28	5.2	0.0
20/26	21	1.16	3.4	0.0
21/28	22	1.04	2.8	0.0

Table12. Results for VUV_RMSVQ (2,3 stage)

Number of bites	Average bit rate	SD	SD > 0.2	SD > 0.4
9/12	9	2.1	25.3	1.00
11/14	11	1.97	18.06	0.64
12/16	12	1.83	13.5	0.44
14/18	14	1.68	10.0	0.30
15/20	16	1.49	7.2	0.2
17/22	18	1.38	6.1	0.17
18/24	19	1.17	5.2	0.0
20/26	21	1.09	3.4	0.0
21/28	22	0.95	2.8	0.0

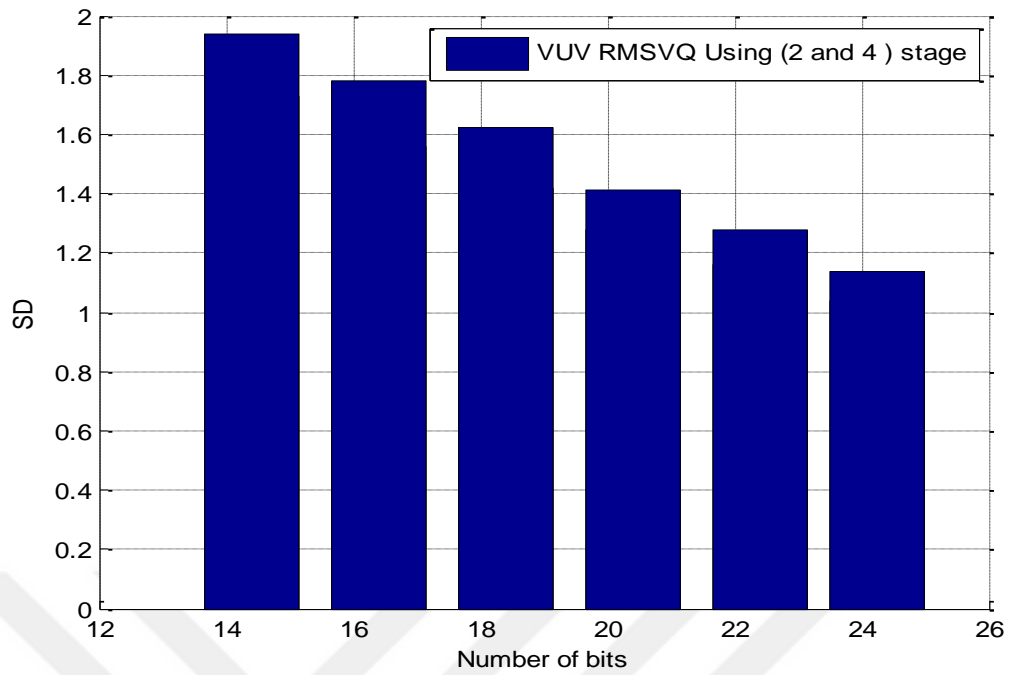


Figure 36. SD for VUV_RMSVQ (2 and 4) stages

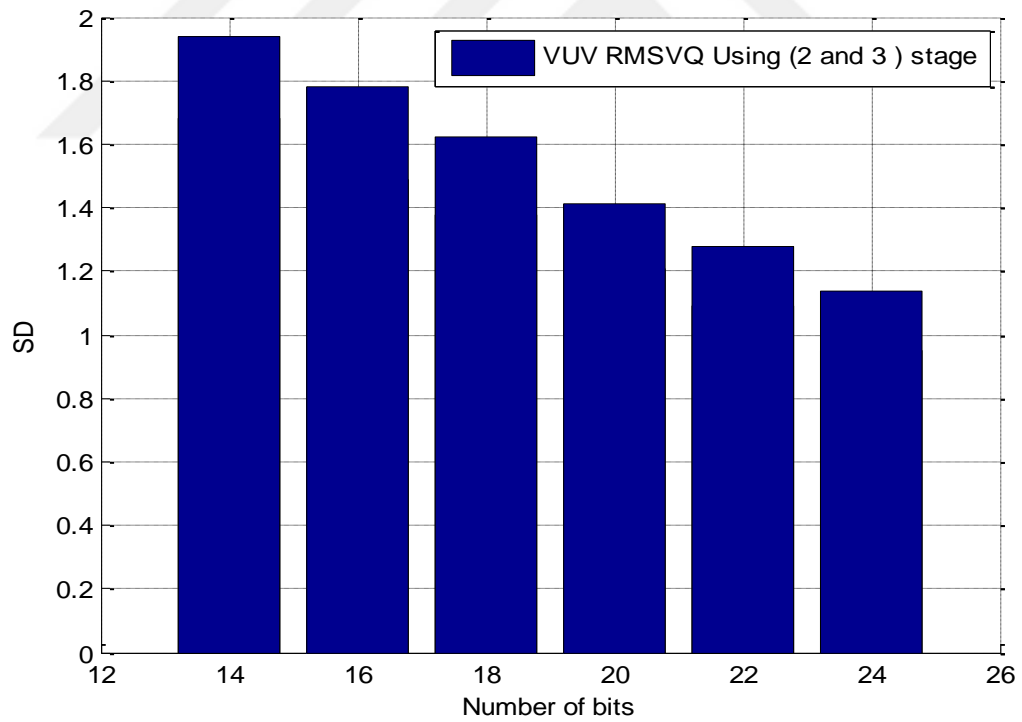


Figure 37. SD for VUV_RMSVQ

Table 13. The SD for noisy speech (5dB)

Number of bites	SD
14	2.56
16	2.33
18	2.16
19	2.09
21	1.93
22	1.87

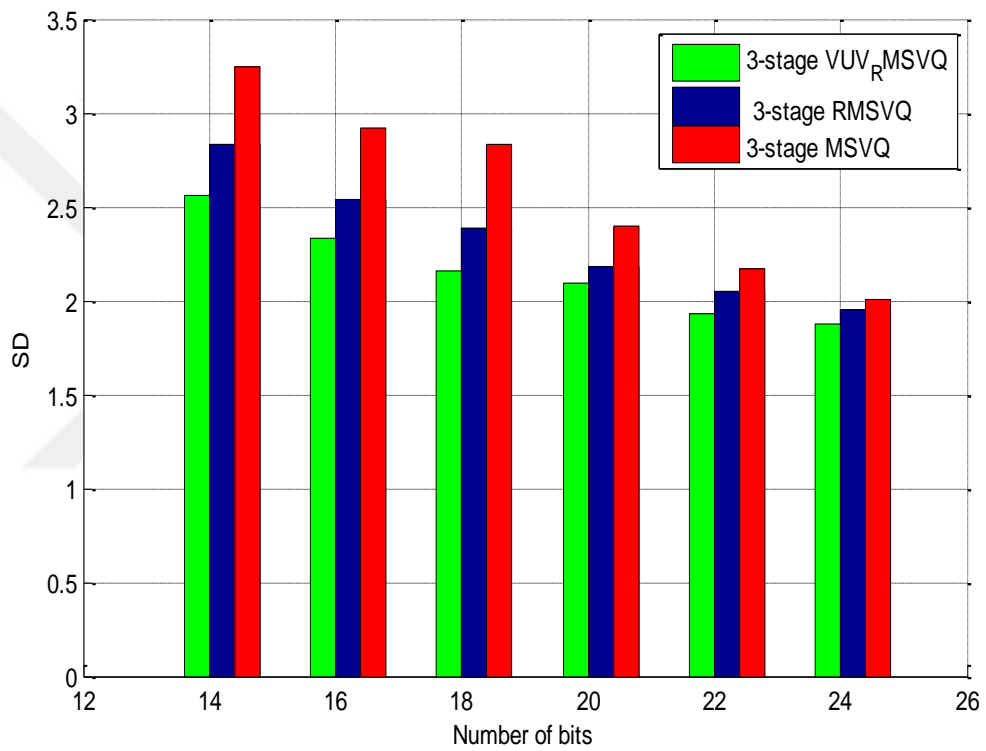


Figure 38. Comparative between MSVQ, RMSVQ and VUV_RMSVQ

Using noisy speech

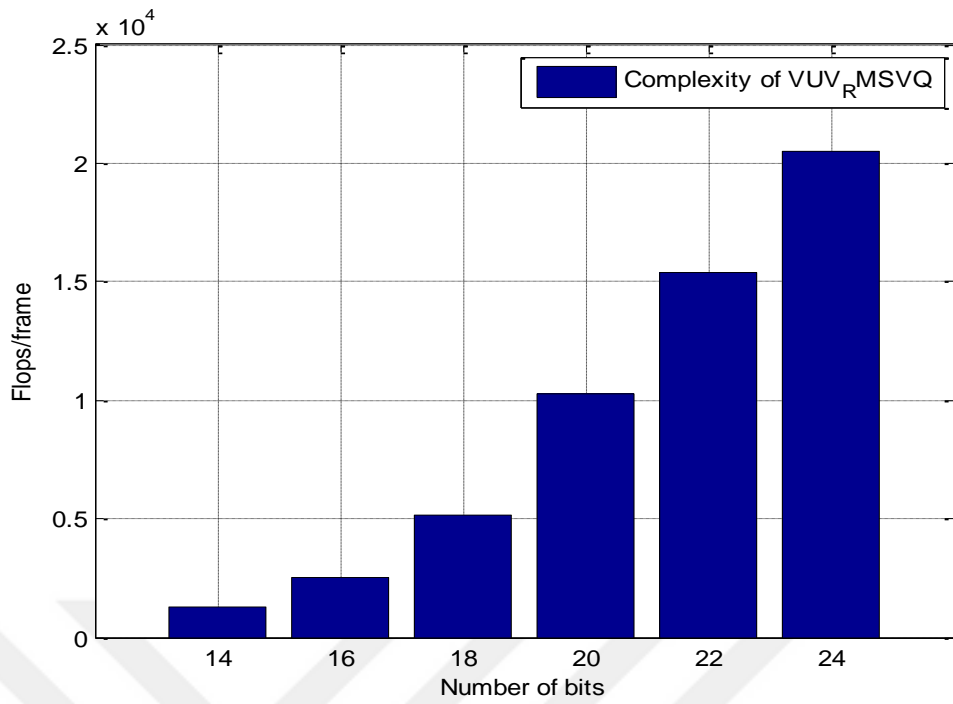


Figure 39. Complexity for VUV_RMSVQ

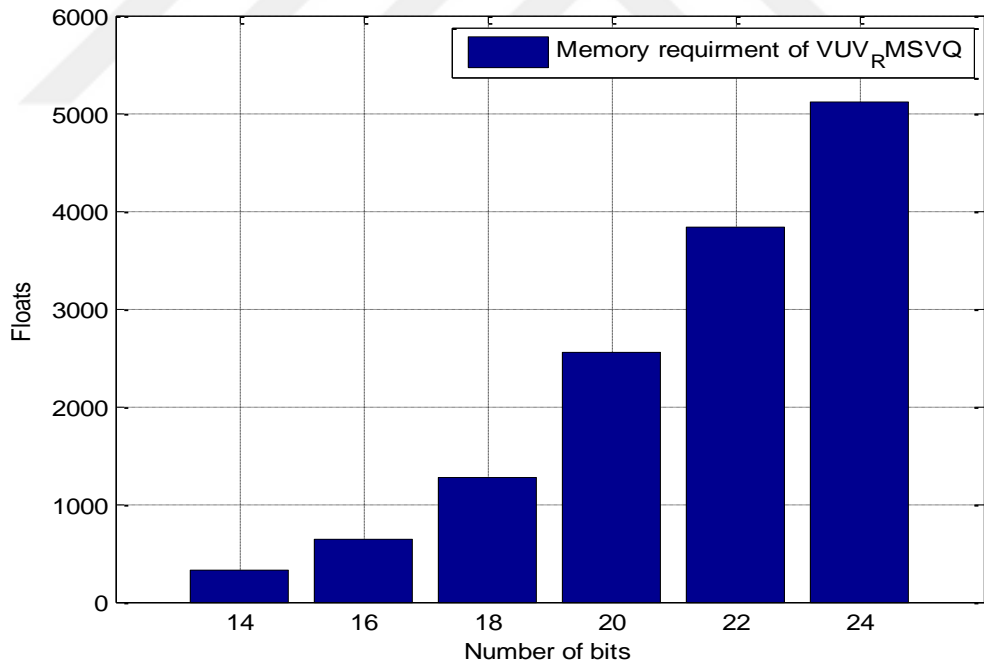


Figure 40. Memory for VUV_RMSVQ

Table14. Results for VUV_RMSVQ (increasing and decreasing number of bits for voiced and unvoiced)

Number of bites	SD	Number of bites	SD
9/12	2.1	12/9	1.88
11/14	1.97	14/11	1.68
12/16	1.83	16/12	1.58
14/18	1.68	18/14	1.42
15/20	1.49	20/15	1.27
17/22	1.38	22/17	1.06
18/24	1.17	24/18	1.00
20/26	1.09	26/20	0.85
21/28	0.95	28/21	0.76

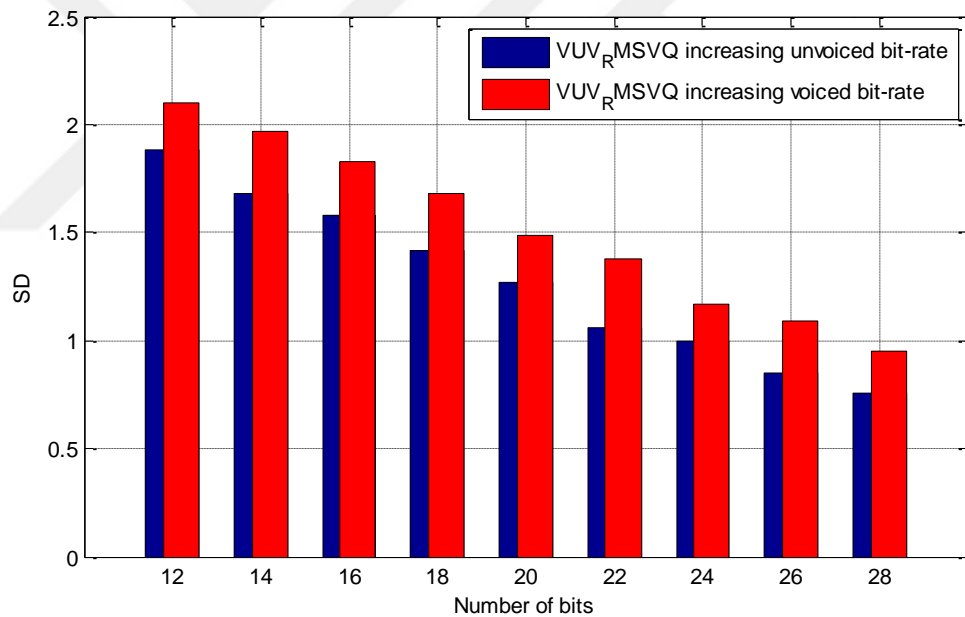


Figure 41. Comparative between increasing bit number for voiced and unvoiced for VUV_RMSVQ

8.7 Performance Analysis

In this section, results are presented for the proposed VUV_RMSVQ scheme. The test data used for these results are separate from the large set of speech data that was used to train the VUV_RMSVQ codebooks and includes speech utterances at different levels and noise cases. The test speech data (greater than 108,000 frames) is passed through the front-end mode classification scheme and quantized LSF vectors are

reconstructed using the VUV_RMSVQ codebooks. The quantized and original LSF vectors are compared using averages and outlier percentages. As we can see from tables and figures and comparing our new method with all previous work, as shown in figure 39 VUV_RMSVQ gave very much improvement when compared with MSVQ and RMSVQ with percentage about 10% improvement by separating the speech into voiced and unvoiced and applying this method on RMSVQ we have increased the SD performance as seen from the figures. We have applied another method, to increase the performance of SD, as seen from Table 14 we have decreased and increased the number of bits for voiced and unvoiced and reversed, from the result we can see the performance will increase and the SD will decrease when the number of bits for unvoiced are higher than the number of bits for the voiced.

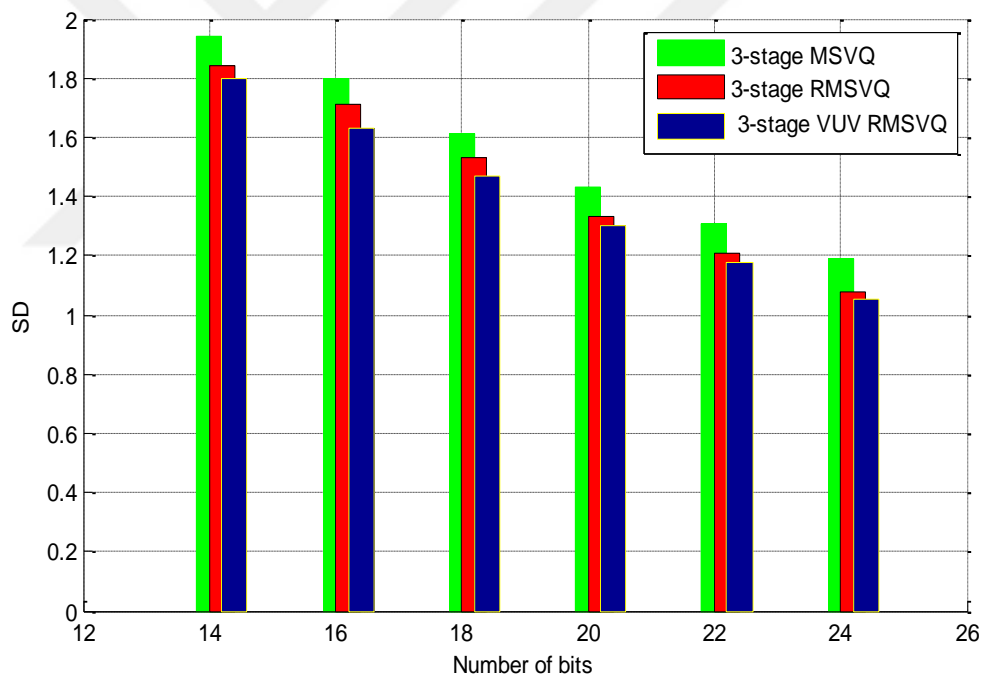


Figure 42. Comparative between MSVQ, RMSVQ and VUV_RMSVQ using clean speech

CHAPTER 9

CONCLUSION AND FUTURE WORK

9.1 Conclusion

The core work of this thesis involves in improving the performance of the vector quantization techniques using hybrid methods. This work is carried out by proposing some hybrid vector quantization techniques and by comparing their performance with the existing vector quantization techniques. The performance of the vector quantization techniques is examined by using linear predictive codefor reducing the bit-rate of a speech signal. The performance of various vector quantization techniques is measured and compared at different bit-rates. After the analysis of the speech signal is made by framing, overlapping, and windowing, calculating the linear predictive coefficients (LPC) and obtaining the residue of the speech signal. Immediately after speech analysis, speech synthesis will be carried to check the quality of the reconstructed speech signal. Vector Quantization is the process that is carried out between speech analysis and synthesis. In speech synthesis the filter parameters are the linear predictive coefficients obtained after vector quantization. The performance of vector quantization is measured using spectral distortion, complexity and memory requirements.

- **The first method proposed in this thesis:** the aim was to find the best vector quantization method with less SD .The performance of two popular quantization methods, SVQ and MSVQ, was evaluated in LSF quantization. To achieve the best performance with MSVQ, it was found that the number of stages should generally be kept at minimum provided that maximum size codebooks are used for each stage. The evaluation of SVQ with 22 different splitting schemes indicated that none of the splitting schemes outperformed the others at all bit rates. The bit allocation was more complicated with SVQ than with MSVQ because the optimal allocation depends on both the sub-vector lengths and the frequency bands that the splits cover. The comparison between the quantization methods indicated that MSVQ clearly outperforms SVQ. By using MSVQ instead of SVQ, at least 2–3 bits can be saved without any quality

degradations. The comparison between the quantization methods indicated that MSVQ clearly outperforms SVQ. By using MSVQ instead of SVQ, at least 2–3 bits can be saved without any quality degradations.

- **The second method proposed in this thesis:** after comparing the results from previous method, we decided to use residual vector quantization and apply it on both (SVQ and MSVQ). A residual split vector quantization (RSVQ) and residual multistage vector quantization (RMSVQ) scheme for sequential quantization of LPC coefficients was presented. When we compare both (RSVQ) and (RMSVQ) as we can see from tables in chapter 7 RMSVQ give best performance than RSVQ. Then we compared RSVQ with SVQ according to the bit allocation from 14 to 24 that RSVQ give better performance than SVQ about 27% improvement, and RMSVQ give lower SD when compared to MSVQ, as we can see from the figures and tables that was given previously RMSVQ give best performance than RSVQ, from this point we have decided to go further using RMSVQ and try to find algorithm that gives better SD.
- **The final method proposed in this thesis:** in this method we decided to separate the speech to voiced and unvoiced using residual multistage vector quantization. The proposed technique VUV_RMSVQ involves speech mode based on MSVQ design using residual LSF vectors obtained from the first-order backward prediction of LSF vectors. It is shown that efficient quantization performance can be obtained by designing a 18-24 bit two-stage codebook for both mode A (Voiced) and mode B (Unvoiced). The performance is compared to RMSVQ and MSVQ, as shown from the graphs VUV_RMSVQ gives very good performance in SD and achieved the three condition of the quantizer which are:

The average or mean of the spectral distortion (SD) must be less than or equal to 1dB.

There must be no outlier frames having a spectral distortion greater than 4dB.

The number of outlier frames between 2 to 4dB must be less than 2%.

REFERENCES

1. **Spanias, A.S., (1994)**, “*Speech coding: a tutorial review*”, Proceedings of the IEEE, vol 82, pp. 1541–1582.
2. **Sayood, K., (1996)**, “*Introduction to data compression*”, Morgan Kaufmann Publishers, San Francisco.
3. **Vaseghi, Saeed V., (2007)**, “*Multimedia Signal Processing Theory and Applications in Speech, Music and Communications*”, John Wiley & Sons Ltd.
4. **Bellamy, John C., (2000)**, “*Digital Telephony*”, John Wiley & Sons, Inc, Wiley Series in Telecommunications and Signal Processing.
5. **Chu, Wai C., (2003)**, “*Speech coding algorithms: foundation and evaluation of standardized coders*”, Wiley-IEEE.
6. **Kleijn, W.B., and Paliwal, K.K., (1995)**, “*An introduction to Speech coding*”, Speech coding and synthesis, Elsevier science, pp. 1-47.
7. **Tribolet, J., Noll, P., McDermott, B., and Crochiere, R., (1978)**, “*A study of complexity and quality of speech waveform coders*”, in: Proc.IEEE Int. Conf. Acoust., Speech, Signal Processing, vol. 3, pp. 586–590.
8. **Atal, B.S., (1982)**, “*Predictive coding of speech at low bit rates*”, IEEE Transactions on Communications, vol 30, Issue 4, pp.600-614.
9. **Kitawaki, N., Nagabuchi, H., and Itoh, K., (1988)**, “*Objective Quality evaluation for low-bit-rate speech coding systems*”, IEEE Journal on Selected Areas in Communications, vol 6, Issue 2, pp. 242–248.
10. **Cox, R.V., Kroon, P., (1996)**, “*Low bit-rate speech coders for multimedia communication*”, IEEE Communications Magazine, Vol 34, Issue 12, pp. 34–41.
11. **Ming Yang, (2004)**, “*Low bit-rate speech coding*”, IEEE Potentials, vol 23, Issue 4, pp. 32–36.
12. **Tremain, Thomas E., (1982)**, “*The government standard linear predictive coding algorithm*”, LPC-10,” Speech Technology Magazine, pp. 40-49.
13. **Atal, B.S., (2006)**, “*The history of linear prediction*”, Signal Processing Magazine, IEEE, vol 23, Issue 2, pp.154 – 161, March.

14. **Lee, C.H., (1988)**, “*On robust linear prediction of speech*,” IEEE Trans. Acoust. Speech Signal Process, vol. 36, Issue 5, pp.642-650.
15. **Makhoul, J., (1975)**, “*Linear prediction: A tutorial review*”, [Proceedings of the IEEE](#), vol 63, Issue 4, pp. 561- 580.
16. **Fant, Gunnar, (1970)**, “*Acoustic Theory of Speech Production*”, Walter de Gruyter.
17. **Rabiner, L., and Schafer, R., (1978)**, “*Digital processing of speech signals*”, Pearson education.
18. **Paget, Richard**, “*Human speech*”, Routledge, 199
19. **Al-Akaidi, Marwan, (2004)**, “*Fractal speech processing*”, Cambridge university press.
20. **Hongwen, Pei, and Fengji, Shen, (1991)**, “*Research and implementation of linear predictive speech analysis and synthesis*”, in: Proc. IEEE Int. Conf. Circuits and Systems, vol. 1, pp. 22–25, June.
21. **R. McAulay, and Quatieri, T., (1986)**, “*Speech analysis/Synthesis based on a sinusoidal representation*”, IEEE Trans. Acoust. Speech Signal Process, vol. 34, Issue 4, pp.744-754, August.
22. **Lee, Chin-Hui, (1987)**, “*Robust linear prediction for speech analysis*”, in: Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, vol. 12, pp. 289–292. April,
23. **Tzeng, F.F., (1990)**, “*An analysis-by-synthesis linear predictive model for narrowband speech coding*”, in: Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, vol. 1, pp. 209–212, April.
24. **Zahorian, S., and Gordy, P., (1983)**, “*Finite impulse response (FIR) filters for speech analysis and synthesis*”, in: Proc. IEEE Int. Conf Acoust., Speech, Signal Processing, vol. 8, pp. 808–811, April.
25. **Yegnanarayana B., and Murthy, P. Satyanarayana, (1996)**, “*Source- system windowing for speech analysis and synthesis*”, IEEE Trans. Speech Audio Processing, vol. 4, Issue 2, pp.133–137, March.
26. **Kondoz, A. M., (2005)**, “*Digital speech: coding for low bit-rate Communication systems*”, John Wiley & Sons, Ltd., June.
27. **Harrington, Jonathan, (1999)**, “*Techniques in speech acoustics*”, Springer.

28. **Barnwell, T., (1981)**, “*Recursive windowing for generating autocorrelation coefficients for LPC analysis*”, IEEE Trans. Acoust. Speech Signal Process, vol. 29, Issue 5, pp.1062-1066. October.
29. **Knorr, S., (1979)**, “*Reliable voiced/unvoiced decision*”, IEEE Trans. Acoust. Speech Signal Process, vol. 27, Issue 3, pp.263-267, June.
30. **Atal, B., and Rabiner, L., (1976)**, “*A Pattern recognition approach to voiced-unvoiced-silence classification with applications to speech recognition*”, IEEE Trans. Acoust. Speech. Signal Process, vol. 24, Issue 3, pp.201-212, June.
31. **Bachu, R.G., Kopparthi, S., Adapa, B., and Barkana, B.D.,** “*Separation of voiced and unvoiced using zero-crossings rate and Energy of the speech signal*”, Electrical Engineering Department, School of Engineering, University of Bridgeport.
32. **Greenwood, Mark, and Kinghorn, Andrew,** “*Suving: automatic silence /unvoiced/voiced classification of speech*”, Department of Computer Science, University of Sheffield, UK.
33. **Maksym, J., (1973)**, “*Real-time pitch extraction by adaptive prediction of the speech waveform*”, IEEE Trans. Audio and Electroacoustics, vol. 21, Issue 3, pp.149–154, June.
34. **Medan, Y., Yair, E., and Chazan, D., (1991)**, “*Super resolution pitch determination of speech signals*”, IEEE Trans. Signal Process, vol. 39, Issue 1, pp.40–48, June.
35. **Ramachandran, R.P., Kabal, P., (1989)**, “*Pitch prediction filters in speech coding*”, IEEE Trans. Acoust. Speech Signal Process, vol. 37, Issue 4, pp.467-478, April.
36. **Sondhi, M., (1968)**, “*New methods of pitch extraction*”, IEEE Trans. Audio and Electroacoustics, vol. 6, Issue 2, pp.262–266, June.
37. **Wise, J., Caprio, J., and Parks, T., (1976)**, “*Maximum likelihood pitch estimation*”, IEEE Trans. Acoust. Speech Signal Process, vol. 24, Issue 5, pp.418-423, October.
38. **Ross, M., Shaffer, H., Cohen, A., Freudberg R., and Manley, H., (1974)**, “*Average magnitude difference function pitch extractor*”, IEEE Trans. Acoust. Speech Signal Process, vol. 22, Issue 5, pp.353-362, October.
39. **Rabiner, L., (1977)**, “*On the use of autocorrelation analysis for pitch detection*”, IEEE Trans. Acoust. Speech Signal Process, Vol. 25, Issue 1, pp.24-33, February.

40. **Rabiner, L., Cheng, M., Rosenberg, A., and McGonegal, C., (1976),** “*A comparative performance study of several pitch detection algorithms*”, IEEE Trans. Acoust. Speech Signal Process, vol. 24, Issue 5, pp.399-418, October.
41. **Goldberg, Randy, and Riek, Lance, (2000),** “*A practical handbook of speech coders*”, CRC press.
42. **Lemmetty, Sami, (1999),** “*Review of speech synthesis technology*”, Master's Thesis, Department of Electrical and Communications Engineering, Helsinki University of Technology, 1999, pp. 1-113, March.
43. **Kang G. and Everett, S., (1984),** “*Improvement of the narrowband LPC synthesis*”, in: Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, vol. 9, Part 1, pp. 25–28, March.
44. **McCree, A. V., Barnwell, T.P., (1991),** “*A new mixed excitation LPC vocoders*”, in: Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, vol. 1, pp. 593–596, April.
45. **Shannon C. E., (1959),** “*Coding theorems for a discrete source with a fidelity criterion*”, IRE National Convention Record, part 4, pp. 142–163.
46. **Gray, R.M., and Neuhoff, D. L., (1998),** “*Quantization*”, IEEE Trans. Inform. Theory, vol. 44, Issue 6, pp. 2325-2383, October.
47. **Makhoul, J., Roucos S., and Gish, H., (1985),** “*Vector quantization in speech coding*”, Proceedings of the IEEE, vol 73, Issue 11, pp. 1551- 1588, November.
48. **Quatieri, Thomas F., (2004),** “*Discrete-Time speech signal processing*”, Low price edition.
49. **Stephen, So. and Paliwal, K. K., (2007),** “*Efficient product code vector quantization using switched split vector quantizer*”, Digital Signal Processing journal, Elsevier, vol. 17, pp.138-171, January.
50. **Soong, F.K. and Juang, B.H., (1984),** “*Line spectrum pair (LSP) and speech data compression*”, Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, vol 9, Issue 1, pp. 37 40, March.
51. **Kabal, P. and Ramachandran, R.P., (1986),** “*The computation of line spectral frequencies using Chebyshev polynomials*”, IEEE Trans. Acoust. Speech Signal Process, vol. 34, Issue 6, pp.1419-1426, December.

52. **Hong, Kook Kim and Hwang, Soo Lee, (1999)**, “*Interlacing properties of line spectrum pair frequencies*”, IEEE Trans. Speech Audio Process, vol. 7, Issue 1, pp.87–91, January.
53. **Kleijn, W.B., Backstrom, T. and Alku P., (2003)**, “*On line spectral frequencies*”, IEEE Signal Processing Letters, vol 10, Issue 3, pp.75-77, March.
54. **Linde, Y., Buzo, A. and Gray, R. M., (1980)**, “*An algorithm for vector quantizer design*”, IEEE Trans. Commun. COM-28, vol. 1, pp.84–95, January.
55. **Buzo, A., Gray, A.H., Gray, R.M. and Markel, J., (1980)**, “*Speech coding based upon vector quantization*”, IEEE Trans. Acoust. Speech Signal Process, vol. 28, Issue 5, pp.562-574, October.
56. **Sugamura, N. and Farvardin, N., (1988)**, “*Quantizer design in LSP speech analysis and synthesis*”, Proc. Int. Conf. Acoust., Speech, Signal Processing, (New York, NY), pp. 398–401.
57. **Gersho, A. and Gray, R.M., (1992)**, “*Vector Quantization and Signal Compression*”, Kluwer Academic Publishers, Dordrecht.
58. **Pan, J. and Fischer, T. R., (1994)**, “*Vector quantization-lattice vector quantization of speech LPC coefficients*”, in: Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, vol.1, pp.513–516, April.
59. **Viswanathan R. and Makhoul, J., (1995)**, “*Quantization properties of transmission parameters in linear predictive systems*”, IEEE Trans. Acoust. Speech Signal Process, vol. 23, Issue 3, pp.309-321, June.
60. **J. Zhou, J., Shoham, Y. and Akansu, A., (1996)**, “*Simple fast vector quantization of the line spectral frequencies*”, in: Proc. ICSLP, vol. 2, pp. 945–948. October.
61. **Subramaniam, A.D. and Rao, B.D., (2003)**, “*PDF optimized parametric vector quantization of speech line spectral frequencies*”, IEEE Trans. Speech Audio Process, vol. 11, Issue 2, pp.130–142, March.
62. **Soong F. K. and Juang, B. H., (1993)**, “*Optimal quantization of LSP parameters*”, IEEE Trans. Speech Audio Process, vol. 1, Issue 1, pp.15–24, January.
63. **Gardner, W. R, and Rao, B. D., (1995)**, “*Theoretical analysis of the high-rate vector quantization of LPC parameters*”, IEEE Trans. Speech Audio Process, vol. 3, Issue 5, pp.367–381. September.

64. **Davidson, G. and Gersho, A., (1986)**, “*Complexity reduction methods for vector excitation coding*”, in: Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, vol. 11, pp. 3055–3058, April.
65. **Gardner, W.R. and Rao, B.D., (1995)**, “*Optimal distortion measures for the high rate vector quantization of LPC parameters*”, in: Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, vol. 1, pp. 752–755, May.
66. **Moller, U., Galicki, M., Baresova, E. and Witte, H., (1998)**, “*An efficient vector quantizer providing globally optimal solutions*”, IEEE Trans. Signal Process, vol. 46, Issue 9, pp.2515–2529, September.
67. **Sabin, J. and Gray, R.M., (1984)**, “*Product code vector quantizers for waveform and voice coding*”, IEEE Trans. Acoust. Speech. Signal Process, vol. 32, Issue 3, pp. 474-488.
68. **Paliwal. K. K. and Atal, B. S., (1993)**, “*Efficient vector quantization of LPC parameters at 24 bits/frame*”, Speech Audio Process, vol.1, pp.3–14, January.
69. **Xydeas C.S. and Papanastasiou, C., (1995)**, “*Efficient coding of LSP parameters using split matrix quantization*”, in: Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, vol. 1, pp 740-743, May.
70. **Norden, F. and Eriksson, T., (2004)**, “*On split quantization of LSF parameters*”, in: Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, vol. 1, pp. I-157–I-160, May.
71. **Juang, B. H. and Gray, Jr, A. H., (1982)**, “*Multiple stage vector quantisation for speech coding*”, in: Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, vol. 7, pp.597-600, May.
72. **Chan, W. Y., Gupta, S. and Gersho, A., (1992)**, “*Enhanced multi stage vector quantization by joint codebook design*”, IEEE Trans. Commun, vol 40, Issue 11, pp.1693 –1697, November.
73. **LeBlanc, W. P., Bhattacharya, B., Mahmoud, S. A. and Cuperman, V., (1993)**, “*Efficient search and design procedures for robust multi-stage VQ for LPC parameters for 4 kb/s speech coding*”, IEEE Trans. Speech Audio Process, vol. 1, Issue 4, pp.373–385. October.

74. **Pan, J., (1996)**, “*Two-stage vector quantization-pyramidal lattice vector quantization and application to speech LSP coding*”, in: Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, vol. 2, pp. 737–740, May.
75. **Krishnan, V., Anderson, D. V. and Truong, K. K., (2004)**, “*Optimal multi stage vector quantization of LPC parameters over noisy channels*”, IEEE Trans. Speech Audio Process, vol. 12, Issue 1, pp.1–8. January.
76. **Kleijn, W. B., (1995)**, “*An Introduction to Speech Coding, in Speech Coding and Synthesis*”, Elsevier, pp.1-47.
77. **McAulay, R. J. and Quatieri, T. F., (1986)**, “*Speech analysis/synthesis based on a sinusoidal representation*”, ASSP, vol. 34, no.4, pp.744-754.
78. **Kroon, P. and Kleijn, W. B., (1995)**, “*Linear Predictive Analysis by Synthesis Coding*”, in Modern Methods of Speech Processing, Chapter 3.
79. **Sugamura, N. and N. Farvardin, (1988)**, “*Quantizer design in LSP speech analysis and synthesis*”, Proc. Int. Conf. Acoust., Speech, Signal Processing, (New York, NY), pp. 398–401.
80. **Kroon, P. and E. F. Deprettere, (1988)**, “*A class of analysis-by-synthesis predictive coders for highquality speech coding at rates between 4.8 and 16 kbit/s*”, IEEE J. Selected Areas Comm, vol. 6, pp. 353–363,
81. **Paliwal, K. K. and B. S. Atal, (1993)**, “*Efficient vector quantization of LPC parameters at 24 bits/frame*”, Speech Audio Process, vol.1, pp.3–14, January.
82. **Paliwal, K. K. and Kleijn, W. B., (1995)**, “*Quantization of LPC parameters*”, Speech Coding and Synthesis, Eds. New York: Elsevier Science, pp. 433–466.
83. **Makhoul, J., Roucos, S. and Gish, H., (1985)**, “*Vector quantization in speech coding*”, Proc. IEEE. Vol. 73, pp. 1551-1588, November.
84. **Paliwal, K. K. and Atal, B. S., (1991)**, “*Efficient vector quantization of LPC parameters at 24 bits/frame*”, in: Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, pp. 661–664

85. **Jong Kwan Lee, Chang D. Yoo, (2003)** “*Wavelet speech enhancement based on voiced/unvoiced decision*”, Korea Advanced Institute of Science and Technology The 32nd International Congress and Exposition on Noise Control Engineering, Jeju International Convention Center, Seogwipo, Korea.
86. **Yair, E. and Zeger, K.,** “*A method to obtain better codebooks for Vector Quantizers*”
87. **Ram, M. Satya Sai, Siddaiah, P., and Latha, M. Madhavi, (2008),** “*Multi Switched Split vector quantizer*”, International Journal of Computer, Information, and Systems Science, and Engineering, Winter, pp. 1, <http://www.waset.org/journals/ijcisse/v2/v2-1-1.pdf>.
88. **Kleijn, W. B., (1995),** “*An Introduction to Speech Coding, in Speech Coding and Synthesis*”, Elsevier, pp.1-47.
89. **Yang, Ming , (2004),** “*Low bit rate speech coding*”, IEEE Potentials, vol 23, Issue 4, pp. 32–36, October.
90. **Barnwell, T., (1981),** “*Recursive windowing for generating autocorrelation coefficients for LPC analysis*”, IEEE Trans. Acoust. Speech Signal Process, vol. 29, Issue 5, pp.1062-1066, October.
91. **Jaber Marvan, (2007),**“*Voice Activity detection Method and Apparatus for voiced/unvoiced decision and Pitch Estimation in a Noisy speech feature extraction*”, United States Patent 20070198251.
92. **Tzeng, F. F., (1990),** “*An analysis-by-synthesis linear predictive model for narrowband speech coding*”, in: Proc. IEEE Int. 7 Conf. Acoust., Speech, Signal Processing, vol. 1, pp. 209–212, April.
93. **Sinervo, Ulpu, Nurminen, Jani, Heikkinen, Ari, and Saarinen, Jukka,** “*Evaluation of split and multistage techniques in 9 LSF quantization*”,
94. **McAulay, R. and Quatieri, T., (1986)** “*Speech analysis/Synthesis based on a sinusoidal representation*”, IEEE Trans. Acoust. 11 Speech Signal Process, vol. 34, Issue 4, pp.744-754. August.

95. **Lee, Chin-Hui, (1987)**, “*Robust linear prediction for speech analysis*”, in: Proc. IEEE Int. Conf. Acoust., Speech, Signal 13 Processing, vol. 12, pp. 289–292, April.
96. **Lee, C.H., (1988)**, “*On robust linear prediction of speech*”, IEEE Trans. Acoust. Speech Signal Process, vol. 36, Issue 5, pp.642-650, May.
97. **Hongwen, Pei and Fengji, Shen, (1991)**, “*Research and implementation of linear predictive speech analysis and synthesis*”, 17 in: Proc. IEEE Int. Conf. Circuits and Systems, vol. 1, pp. 22–25, June.
98. **Kondoz, A. M., (2005)**, “*Digital Speech: Coding for Low Bit rate Communication Systems*”, John Wiley & Sons, Ltd., June.
99. **Kitawaki, N., Nagabuchi, H. and Itoh, K., (1988)**, “*Objective quality evaluation for low-bit rate speech coding systems*”, 21 IEEE Journal on Selected Areas in Communications, vol 6, Issue 2, pp. 242–248, February.
100. **Kanawade, P. R., Gundal, S. S., (2017)**, “*Tree structured vector quantization based technique for speech compression*”, International Conference on Data Management, Analytics and Innovation (ICDMAI).
101. **Shafi, M., Makwana, V., Nandurbar, A. B. and Parmar, K. R., (2014)**, “*Speech compression using tree structured vector 25 quantization*”, 2nd International Conference on Device,Circuit and Systems(ICDCS).
102. **Tribolet, J., Noll, P., McDermott, B. and Crochiere, R., (1978)**, “*A study of complexity and quality of speech waveform 27 coders*”, in: Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, vol. 3, pp. 586–590, April.
103. **Stephen, So. and Paliwal, K. K., (2007)**, “*Efficient product code vector quantization using switched split vector quantizer*”, Digital Signal Processing journal, Elsevier, vol. 17, pp.138-171, January.
104. **Soong F.K. and Juang B.H., (1984)**, “*Line spectrum pair (LSP) and speech data compression*”, Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, vol 9, Issue 1, pp. 37 40, March.

105. **Kabal P. and Ramachandran R.P., (1986)**, “*The computation of line spectral frequencies using Chebyshev polynomials*”, IEEE Trans. Acoust. Speech Signal Process, vol. 34, Issue 6, pp.1419-1426, December.
106. **Hong ,Kook Kim and Hwang, Soo Lee, (1999)**, “*Interlacing properties of line spectrum pair frequencies*”, IEEE Trans. Speech Audio Process, vol. 7, Issue 1, pp.87–91, January.
107. **Kleijn, W.B., Backstrom, T. and Alku, P., (2003)**, “*On line spectral frequencies*”, IEEE Signal Processing Letters, vol 10, Issue 3, pp.75-77, March.
108. **Linde, Y., Buzo A. and Gray, R. M., (1980)**, “*An algorithm for vector quantizer design*”, IEEE Trans. Commun. COM-28, vol. 1, pp.84–95, January.
109. **Buzo, A., Gray, A.H., Gray ,R.M. and Markel, J., (1980)**, “*Speech coding based upon vector quantization*”, IEEE Trans. Acoust. Speech Signal Process, vol. 28, Issue 5, pp.562-574.
110. **Sugamura, N. and Farvardin, N., (1988)**, “*Quantizer design in LSP speech analysis and synthesis*”, Proc. Int. Conf. Acoust., Speech, Signal Processing, (New York, NY), pp. 398–401.
111. **Gersho A.and Gray R.M., (1992)**, “*Vector Quantization and Signal Compression*”, Kluwer Academic Publishers, Dordrecht.
112. **LeBlanc P., Bhattacharya B., Mahmoud S. A., Cuperman V., (October 1993)**, “*Efficient Search and Design Procedures for Robust Multi-Stage VQ of LPC Parameters for 4 kb/s Speech Coding,*” IEEE Transactions on Speech and Audio Processing, Vol. 1, No. 4.
113. **A. Balwant Sonkamble, D.D.Doye (2012)**, “*Speech Recognition using vector Quantization through modified K-mean LBG Algorithm*”, computer engineering and intelligent System.
114. **Manhar KAVYA, B Premanand (2012)**, “*Comparative Study on vector quantization codebook generation algorithm for wideband speech coding*”,
115. **Selma Ozaydin, Buyurman Baykal.,(2012)**, “*A 1200 bps SPEECH CODER WITH LSF MATRIX QUANTIZATION*”,IEEE international conference,2,667-680.

- 116. Ya Li, Xiaoqun Zhao (2015),** “*vector quantization for LSF coding and codebook storage*”,International conference on communications and networking in china,pp.187.
- 117. Hanwei Wu, Qiwen Wang, and Markus Flierl., (2017),**“*Tree-Structured Vector Quantization for Similarity Queries*” Data Compression Conference
- 118. P. R. Kanawad, S. S. Gundal.,(2017),** “*Tree Structured Vector Quantization Based Technique for Speech Compression*”, International Conference on Data Management, Analytics and Innovation (ICDMAI) Zeal Education Society, Pune, India.
- 119. Pradeep Kumar Shah, Rajendra Prashad Pandey, Rajeev Kumar.,(2016),** “*Vector Quantization with Codebook and Index Compression*”, 5th International Conference on System Modeling & Advancement in Research Trends, India