On the Design of Robust Vector Quantizers for Sources and Channels with Memory

by

Pradeepa Yahampath

A dissertation submitted to The Faculty of Graduate Studies

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Electrical and Computer Engineering

The University of Manitoba

Winnipeg, Canada

November 2001

0-612-79915-8

THE UNIVERSITY OF MANITOBA

FACULTY OF GRADUATE STUDIES

*****

COPYRIGHT PERMISSION PAGE

On the Design of Robust Vector Quantizers for Sources and Channels with Memory

BY

Pradeepa Yahampath

A Thesis/Practicum submitted to the Faculty of Graduate Studies of The University

of Manitoba in partial fulfillment of the requirements of the degree

of

DOCTOR OF PHILOSOPHY

To my parents.

Abstract

Recently, robust quantization has received considerable attention, particularly as a potential approach to joint source-channel coding. This dissertation investigates the practical design of channel optimized vector quantizers (COVQ). Specific problems considered here include COVQs *with memory* and COVQs operating over *channels with memory*. In this work, the emphasis is placed on *soft-decoding* at the receiver.

Vector quantizers with memory are an effective means of quantizing correlated signals. However, when designed without regard to channel errors, these quantizers suffer from degradation of performance due to the propagation of channel errors at the receiver. We consider two important examples of such quantizers, namely, *predictive vector quantizers* (PVQ) and *finite-state vector quantizers* (FSVQ). In the case of PVQ, an iterative algorithm is developed for jointly optimizing the quantizer and the associated linear predictor to a given channel. According to the simulation results presented here, the proposed PVQ designs based on hard-decoding perform comparably to those obtained by a previously studied gradient-search optimization algorithm. Furthermore, it is demonstrated that PVQs with soft-decoding can provide a significant improvement over hard-decoding systems. In the case of FSVQ, a time-recursive decoding algorithm, which exhibits graceful degradation of performance with increasing channel noise, is introduced. Design of channel optimized FSVQ is also considered. Simulation results are presented, which demonstrate that proposed channel optimized FSVQs outperform the memoryless COVQs operating at the same rate.

Finally, in the context of channels with memory, joint equalization and soft-decoding using a sliding-block decoder is investigated. This decoder is a non-linear time-invariant filter based on minimum mean square error criterion. As a practical implementation, multi-layer perceptron (MLP) is considered. Simulation results indicate that MLP-based soft-decoder outperforms a previously studied recursive soft-decoder, particularly under high channel noise. However, the complexity of the optimal sliding-block decoder function is found to increase with the encoder resolution, making the estimation task harder. In an encouraging development, it is shown that the optimal sliding-block decoder for the Gaussian channel approximates a linear function as the channel becomes noisier. Experimental results seem to support this theoretical result.

Acknowledgements

I would like to express deep gratitude to my advisor, Professor Mirek Pawlak, for his invaluable guidance and support, and for sharing his wisdom with me. Working with him has certainly been one of the richest learning experiences in my life. I also wish to thank Professor Ed Shwedyk for taking time to answer my questions on communication theory on many occasions. My sincere thanks are due to Dr. Udaya Annakkage for reading through the manuscript and for his constructive suggestions for improving it. I wish to acknowledge the financial support from NSERC of Canada, University of Manitoba, and TRLabs of Winnipeg. I would also like to thank my friends for their support in numerous ways, and most importantly for their humor. Lunch and coffee breaks with them undoubtedly were some of the best times at school.

Finally, this work would not have been a reality without the love and support of my dear family. I am specially greatful to my wonderful parents for all their effort and for having faith in me. I owe much to my daughter Delanie for being so sweet and for forgiving me for spending long hours at school. The biggest *thank you* goes to my wife Ioni for her incessant support, understanding, and patience all these years.

Table of Contents

List of Figures

List of Tables

List of Acronyms

AR	auto-regressive
AWGN	additive white Gaussian noise
BER	bit error rate
BSC	binary symmetric channel
COFSVQ	channel optimized FSVQ
COPVQ	channel optimized PVQ
COVQ	channel optimized vector quantization
CSNR	channel signal-to-noise ratio
dB	deci-Bell
DMC	discrete memoryless channel
DPCM	differential pulse code modulation
FSVQ	finite-state vector quantization/quantizer
G-M	Gauss-Markov (source)
GLA	generalized Lloyd algorithm
IA	index assignment
iid	independantly and identically distributed
ISI	inter-symbol interference
JSC	joint source-channel coding
LPC	linear predictive coding
LS-FSVQ	labeled-state FSVQ
LT-FSVQ	labeled-transition FSVQ
MA	moving-average
MAP	maximum a posteriori probability
MLP	multi-layer perceptron
MMSE	minimum mean square error
MSE	mean square error
MSVQ	multi-stage vector quantization
OLT-FSVQ	omniscient, labeled transition FSVQ
PCM	pulse code modulation
pdf	probability density function

PVQ	predictive vector quantization/quantizer
R-SD	recursive soft-decoder
SB-SD	sliding-block soft-decoder
SNR	signal-to-noise ratio
SQ	scalar quantization/quantizer
TSVQ	tree-structured vector quantization
VA	Viterbi algorithm
VA-HD	Viterbi algorithm based hard decoder
VQ	vector quantization/quantizer

Chapter 1

Introduction

Explosive growth of wireless networks, the Internet, and other multi-media based services over public networks in recent years has made the efficient use of narrow-band digital channels a very important issue. A large portion of traffic in these communication systems involves data, derived from analog signals such as speech, audio, and video. It is not only required that these analog signals be coded into digital data, but also that the bit-rate of the coded representation be low enough to meet the bandwidth requirements of the channel. The process of obtaining a coded representation for an information bearing signal is referred to as *signal compression* or *source coding*. Fundamentally, source coding requires the elimination of any *redundancy* present in the signal to be coded. When the signal is analog, source coding is necessarily *lossy* in that, some *distortion* is introduced into the signal by the coding process. Thus, the goal of source coding is either to minimize the bit rate for a given level of average distortion or to minimize the average distortion for a given bit rate. There are two essential steps in coding an analog signal into a digital data stream. First is *sampling*, which converts the continuous-time signal into discrete samples.

Second is *quantization*, by which the analog samples[1] are mapped onto a finite set of values, which can be represented by a digital code. In *scalar quantization* (SQ), a continuous-valued variable (*i.e.*, a signal sample) is approximated by a value from a predetermined finite set of values. *Vector quantization* (VQ) is the generalization of scalar quantization to vectors in that, a vector-valued variable is mapped onto a finite set of vectors. Theoretically, VQ of a set of variables is always better than individual SQ of the same variables, and in practice VQ can result in a considerable gain over SQ. In recent years, VQ has received a great deal of attention as a signal compression method [1], [2], [3], and has been considered for many applications, including speech and image coding [4], [5].

Another issue that has to be addressed in digital transmission is *channel coding*, the objective of which is to mitigate the effects of channel distortion on the data received at the channel output. In order to reliably send data over a noisy channel, they must be coded using a redundant code prior to the transmission, so that not all channel outputs are valid codewords. The widely used approach to designing a digital transmission system is to treat source coding and channel coding problems separately. That is, the source code is designed by considering the properties of the source alone, while the channel code is designed by considering the properties of the channel alone. According to the well known source-channel separation theorem due to Shannon [6], there is no loss of performance in designing source code and channel code separately, if infinitely long sequence of source vectors are used in VQ and infinitely long codewords are used in channel coding (*i.e.*, optimal source code and optimal channel code can be combined to obtain the optimal source-channel code). However, these conditions cannot be met in practical designs due to obvious restrictions on computational complexity and allowable delays. Furthermore, the

[1]It is common to transform signal samples into other representations prior to quantization.

separation theorem is not necessarily valid for scenarios encountered in widely used multi-user systems, packet-networks, and wireless systems. Hence, the optimality of separately designed source codes and channel codes remains questionable in many practical situations. In *joint source-channel coding* (JSC), the design of source and channel codes are combined in some manner. JSC can take many different forms and many such methods tend to be application specific, see [7] and references cited therein. While in practice, neither separate designs nor a joint design may yield the optimal solution to the given problem, for a given implementational complexity (and coding delay) the joint design may yield better performance in some applications.

The main focus of this dissertation is a JSC method in which a vector quantizer is designed to minimize an end-to-end distortion measure that includes the distortion due to channel. As the resulting code acts as a source code which is robust against channel distortion, the need for an explicit channel code is eliminated and the available channel bandwidth is better utilized for transmitting source information. This approach is known as *channel optimized vector quantization* (COVQ). We propose and investigate several new algorithms for designing of COVQ.

1.1 Background

The concepts of lossy coding and vector quantization were introduced by Shannon in his classic papers [6] and [8]. In particular, in [8] he proved that there exists a lower bound to the rate $R(D)$, at which an independently and identically distributed (iid) source can be coded to achieve a given value D of an additive distortion measure. Achieving the lower bound requires block source coding (vector quantization). $R(D)$, called the *Shannon rate-distortion function*, is a function of the source probability density and the distortion measure, and represents an asymptotical upper bound to the performance of any VQ, which may be achieved only in the limit of infinite block length (vector dimension). Shannon's results have also been generalized to sources with memory, see [9] for a historical account and a review of rate-distortion theory. Shannon's theory however does not show how optimal VQ can be designed in practice, and in general the difficult problem of finding the optimal vector quantizer remains unsolved. In terms of practical design, perhaps the most significant contribution came from the work of Lloyd [10] [2]. He derived the necessary conditions for a fixed-rate scalar quantizer to be optimal under the square error distortion measure. These conditions are now widely known as *Lloyd optimality conditions*[3]. Based on these results, Lloyd introduced two iterative algorithms, called "Method I" and "Method II" for designing locally optimal quantizers (*i.e.*, only correspond to a local minimum of error function) for a given source, characterized by its density function[4]. Subsequently, Linde *et al.* [12] generalized Method I to the empirical design of vector quantizers. This algorithm, now widely known as *generalized Lloyd algorithm* (GLA) or *Linde-Buzo-Gray* (LBG) algorithm, uses a set of sample vectors obtained from a source to iteratively design a locally optimal VQ.

The design of optimal quantizers for noisy channels was first studied by Kurtenbach and Wintz [13]. The basic difference between their work and that described above is the inclusion of channel errors in the distortion measure. They derived the quantization points and the transition levels of the scalar quantizer which minimizes the mean-square error for a given source density function and a channel transition matrix. As the resulting set of equations were not explicitly solvable, they used

[2]The work initially appeared in an internal report of the Bell Laboratories and was presented in parts at the Institute of Mathematical Statistics meeting in 1957, and was subsequently published in 1982.

[3]Interestingly, Lukaszewicz and Steinhaus also have previously derived similar conditions in a different context, see [2].

[4]Max [11] also independently discovered Method II in 1960.

an iterative procedure based on the Max's algorithm [11] to obtain a locally optimal solution. Subsequently, the problem was further investigated by Farvardin and Vaishampayan in [14]. They identified that, in order to ensure the convergence of the iterative algorithm, it is necessary to impose the constraint $T_{i-1} \leq T_i$, where T_{i-1} and T_i are transition levels that define the i^{th} quantization interval. This situation does not arise in the case of noise-free channel [5]. Ayanoglu and Gray [15] considered the design of trellis waveform coders for noisy channels and provided an iterative codebook improvement algorithm. Trellis waveform coding is a block coding method in which the decoder is a finite-state machine and the encoder is a search algorithm matched to the decoder.

The problem of optimizing a vector quantizer to both source and channel was first addressed by Kumazawa *et al.*, [16]. They derived the necessary conditions for the optimality of encoder and decoder in the presence of a noisy channel and employed those conditions in GLA to design locally optimal vector quantizers. A more complete treatment of *channel-optimized vector quantization* (COVQ) was later provided by Farvardin and Vaishampayan [17]. In particular, they showed that encoding regions in a COVQ are convex polytopes and that the complexity of encoding was not worse than in ordinary VQ. Several authors have reported extensions of COVQ to specific VQ structures. In [18], similar ideas were used to design shape-gain VQ (SGVQ) for noisy channels and the technique was applied to robust image coding. In experimental comparisons with ordinary SGVQ, the channel optimized SGVQ showed improved performance in the presence of channel noise, while computational complexities were reported to be comparable. Phamdo *et al* [19] applied the idea of COVQ to design tree-structured VQ (TSVQ) and multi-stage VQ (MSVQ) for noisy channels and reported substantial performance improvements over ordinary TSVQ

[5] Note also that such constraint is not explicitly required in design based on training sequences.

and MSVQ in coding Gauss-Markov sources under high channel noise. Hussain *et al.*, [20], [21] developed algorithms for designing finite-state VQ (FSVQ) for noisy channels and considered applications in speech coding. Lindén [22], [23] investigated predictive VQ (PVQ) for noisy channels and proposed gradient descent algorithms for optimizing the predictor and decoder codebook to a noisy channel. Both FSVQ and PVQ are adversely affected by channel noise, since they have a feedback structure in the decoder. A more detailed review of previous work on PVQ and FSVQ for noisy channels is presented in Section 1.2.

In the work mentioned so far, a discrete memoryless channel (DMC) was assumed and the channel was characterized by its transition probability matrix. Even though many practical channels can be accurately modeled as a DMC, the actual signals used for conveying discrete symbols across a channel are analog. In order to obtain a discrete output at the receiver, a *detector* is used on the analog channel outputs. However, when the objective is to minimize the mean square error in reconstructing a continuous signal, decoding based on discretized channel outputs can be viewed as a sub-optimal solution to the underlying MMSE estimation problem. The optimal solution is to perform estimation based on continuous channel outputs. This approach is referred to as *soft decoding* as opposed to *hard decoding* used with the discrete channel model. Essentially, soft VQ decoding leads to an improvement in performance over hard VQ decoding, as there is a loss of information due to detection process in the latter. Experimental results reported in this thesis, as well as in the literature, demonstrate that a significant performance gain can be achieved by using soft-decoding in COVQ.

In [24], Vaishampayan and Farvardin considered the problem of jointly optimizing the vector quantizer and the modulation signal set for waveform channels in which the average transmitter power was limited. In their formulation, the decoder was constrained to be a linear mapping from continuous channel output space to the source signal space. Performance comparisons were made against a system with ordinary VQ (designed by GLA) and a maximum likelihood detector. Even though the linear decoder makes analysis tractable, it is clearly suboptimal for reconstructing a signal subjected to the non-linear process of quantization. Liu *et al.*, [25] considered optimum soft-decoding without the linearity constraint for the additive Gaussian noise channel (AWGN). While the conditions for optimality of the encoder and decoder were generalizations of those in [16] and [17] for the discrete channel model, the optimization of signal constellation was performed using a gradient search. Subsequently in [26], the same authors investigated a sequential decoding algorithm for Rayleigh fading channel using a non-linear soft-decoder. Skoglund and Hedelin [27] proposed an interesting implementation of the optimal soft-decoder based on Hadamard matrix and considered the application to binary channels. The Hadamard matrix can be used to express the decoder output vector in terms of the individual bits in the received codewords. In a related work, Skoglund and Ottosson [28] studied multi-user soft-decoding in the context of CDMA[6]. Phamdo and Alajaji [29], [30] investigated soft-decoding for VQ based on soft-decision demodulators for binary Gaussian and Rayleigh fading channels. A soft-decision demodulator discretized the channel output to a higher resolution than the hard-decision demodulator. It was demonstrated experimentally that the performance of the resulting COVQ approached that of a soft-decoder based on continuous channel outputs, when the demodulator resolution is increased. This however comes at the cost of increased storage requirements at the decoder. Recently, soft-decoding has shown to be particularly useful on channels with

[6]In code-division multiple-access (CDMA) several sources (users) are simultaneously transmitted over a shared channel.

intersymbol interference, where the soft-decoder can be used as a combined equalizer and a VQ decoder [31], [32], [33], [34]. This will be further discussed in Section 1.2.

Most work on COVQ assumes a known, stationary channel. This assumption however may not be valid in a number of situations. Wang [35] studied COVQ for time varying finite-state Markov channels. The basic idea is to design a separate COVQ for each channel state having a known signal-to-noise ratio (channel state). Models are derived for meteor burst channels and Rayleigh fading channels, and iterative algorithms are proposed for the design of COVQs for these channels. In a related work, Duman and Salehi [36] considered the design of scalar quantizers when only noisy information about the channel state is available to the encoder and decoder. Jafarkhani and Farvardin [37] also investigated the design of COVQ under channel variations. In particular, they considered the design of COVQ, when the pdf of channel bit-error rate is known.

A problem closely related to COVQ is the optimal channel codeword assignment or *index assignment* (IA) for a VQ operating over a noisy channel. The problem here is to assign channel codewords to the output of a given VQ encoder (typically designed for a noise free channel) so as to minimize the average distortion over a noisy channel. Intuitively, it is clear that one can reduce the average distortion by assigning binary channel codewords that are close in Hamming distance to code vectors in the VQ codebook that are close in distortion measure of the quantizer (*e.g.*, Euclidean distance). This problem can also be viewed as a special case of COVQ in which the encoder partition is fixed. As the mapping to be optimized in this case is one from a set of integers (encoder output) to the set of channel codewords, best IA selection is a combinatorial optimization problem. Clearly, the exhaustive search for the globally optimal solution is intractable in most but simplest cases. On the other hand heuristic search algorithms have been studied and found to be effective [38], [39], [40]. In [39], Zeger and Gersho considered an algorithm which iteratively switches positions of two vectors in the codebook to assure a monotone decrease in average distortion. Farvardin [40] investigated the use of simulated annealing. The general conclusion here is that substantial improvements in performance can be achieved when IA is chosen to match the channel noise level, rather than arbitrarily.

Theoretical studies of COVQ appear sparsely in the literature. In [41], a study of asymptotically optimal noisy channel VQ is presented. In [42], the convergence of empirical error of noisy channel VQ is studied. It is shown that, in terms of the convergence rate (as a function of training set size), the design of noisy channel vector quantizers is not harder than the design of quantizers for noise free channels.

Finally, it is worth mentioning that methods other than GLA have also been investigated for both VQ and COVQ design, see for example [43].

1.2 Related Work

Feedback VQ

A VQ in which the output depends solely on the current input is said to be *memoryless.* According to Shannon's theory on rate-distortion coding of continuous sources, memoryless VQ is sufficient to achieve near optimal performance, if one is prepared to use arbitrary long vectors, *i.e.*, dimension $d \to \infty$. However, practical limits to vector dimension exist, particularly due to the fact that the complexity of a rate R encoder grows as $O(2^{Rd})$. When the source is correlated, a VQ having *memory* can be used to obtain better performance with small vector dimensions. In a VQ with memory, the encoder output depends not only on the current input, but also on the previous inputs. This is usually achieved by employing feedback in the encoder and decoder. Subsequently, such a quantizer has a time varying encoder partition and a decoder codebook. While many quantization methods exist which utilize feedback or memory, predictive VQ (PVQ) and finite-state VQ (FSVQ) have received considerable attention in both speech and image coding [3], [44]. PVQ in particular has been considered for linear-predictive coding (LPC) of speech. In LPC, typically a vector of coefficients representing the short-term spectrum of the speech signal is extracted and encoded. These vectors usually exhibit a high inter-vector correlation which can be exploited with predictive coding. The basic principle behind feedback VQ is to quantize a given vector based on the previously observed vectors, *i.e.*, history of the input signal. Since both encoder (on transmitter side) and decoder (on receiver side) must be able to observe the same history, this information is derived from the reconstructed signal rather than the original signal, thus introducing feedback in both encoder and decoder. A major drawback of feedback VQ is the propagation of errors in the decoder in the presence of channel noise; an error in a single channel output leads to a sequence of erroneous outputs from the decoder. In order to avoid performance degradations in such situations, encoder and decoder design must take channel errors into account. Recent studies on noisy channel PVQ and FSVQ have shown that significant improvements in performance can be obtained by redesigning these systems to minimize error propagation [44], [22], [45], [20].

PVQ [3], [46] is the vector extension of scalar differential pulse code modulation (DPCM), in which the error resulting from predicting the input vector is quantized. When the prediction is good, the error variance is much lower than the signal variance, resulting in an overall coding gain relative to direct coding of input vector [3]. As mentioned above, the prediction at both encoder and decoder is based on the reconstructed signal, and hence we have feedback paths through the predictors at both ends. In [47], the joint optimization of quantizer, predictor (linear), and sampling rate of a DPCM system was studied. Among other things, they suggested the optimization of quantizer and predictor in an iterative approach resembling Lloyd algorithm. They derived analytical equations describing the optimal values for the coefficients of linear predictor and the levels and intervals of the quantizer for prediction error. The solution of these equations requires numerical methods. It was shown that, by decreasing the prediction gain from that of optimal linear predictor, it is possible to improve the noisy channel performance of a DPCM system. More importantly, it was shown that, contrary to popular belief, the effect of channel errors on overall MSE is no more serious in DPCM than in PCM. However, we note that the effect of channel errors relative to quantization errors is much more serious in DPCM. A discussion on noisy channel performance of DPCM can also be found in [48]. More recent work on approaches to robust PVQ can be found in [44]. In [22], Lindén investigated gradient descent algorithms for optimizing linear prediction-based PVQ to a DMC. In these algorithms, an initially chosen codebook (for prediction error) and a predictor are updated in a direction which will decrease the average distortion, while optimal encoder partition is defined in terms of the decoder and channel. Simulation results are reported for blocked Gauss-Markov processes as well as for line spectral frequency (LSF) vectors of speech signals, which indicate that the predictive codes designed using proposed *channel optimized PVQ* (COPVQ) algorithms are much superior to memoryless COVQ. Typically 1-2 dB increase in SNR over memoryless COVQ has been achieved for Gauss-Markov source at 5% channel bit error rate, with gain being higher for smaller vector dimensions.

In FSVQ [3], [49], both encoder and decoder are finite-state machines with a different quantizer associated with each state and state transitions determined by the history of the observed signal. However, in order that both encoder and decoder be able to trace the same sequence of states, state transitions are actually effected by feeding back the quantizer output. The effect of channel noise is much more disastrous on FSVQ than on PVQ. A channel error usually 'derails' the decoder from following the same state sequence as the encoder, which in effect causes the decoder output to virtually become random. Hence, FSVQ shows a dramatic degradation in performance as the channel becomes noisy (see Fig. 4.2). Hussain *et. al*, [20], [21] has studied the design of FSVQ for noisy channels and considered two approaches. In the first, the encoder state is protected by channel coding and explicitly transmitted, so that the decoder need not be a state machine. In order to reduce the additional overhead due to this, the encoder state is transmitted only periodically. The missing states are then estimated using a maximum a posterior sequence detection procedure which requires a delay equal to the intervals at which the encoder state is transmitted. According to the simulation results reported in [20], this method appears to be effective only when the channel is quite noisy. Furthermore it requires a decoding delay, which can be objectionable in applications such as speech coding for which FSVQ is a strong candidate. In the second approach, an FSVQ with a restricted next-state function is designed so that the next-state is solely determined by the previous output of the encoder. With this next-state function, an error in a received channel codeword affects only the following state and, upon receiving a correct channel codeword, the decoder returns to the correct state. This restricted next-state rule, though sub-optimal at very low channel noise levels, results in superior performance compared to the first approach as well as to memoryless COVQ, when the channel error rate is increased.

Soft-decoding VQ for Channels with Memory

In digital communication systems, limited bandwidth and non-linear characteristics of various circuit components cause signals transmitted during adjacent symbol intervals to interfere with each other. This is commonly known as *intersymbol interference* (ISI). The traditional approaches to dealing with ISI include interleaving, maximum likelihood sequence detection (Viterbi decoding) and linear or non-linear equalization filtering. While the use of such techniques allows one to treat the channel as being memoryless in the design of VQ decoders, recent research has shown that considerable performance gains can be achieved by using soft-decoders that operate on the output of ISI channels directly. In [31], Kafedziski and Morell investigated the use of the recursive a posteriori probability estimation algorithm of [50] for soft VQ decoding over linear Gaussian channels. In the terminology of estimation theory, soft-decoding of this type can be considered as *fixed-lag smoothing* [51]. Subsequently, Skoglund [32], further investigated the same approach. In his formulation, encoder statistics were also considered in the decoder expression to utilize the *residual redundancy* [52] of transmitted data for error protection. According to the experimental results reported in [32], the proposed soft-decoding algorithm yields a considerable improvement over a hard decoding scheme involving Viterbi equalization, in the presence of severe ISI. However, a major problem with the said algorithm is that it involves the evaluation of a sum with a number of terms that grows exponentially with the encoder resolution. In [32], Skoglund also investigated approximate computation of the desired sum via the generalized Viterbi algorithm. With this approach the computational complexity of the algorithm can be traded-off for performance. Another sub-optimal approach was also considered in [33].

1.3 Dissertation Outline

This dissertation is concerned with algorithms for practical design of noisy channel VQ. Two main problems have been addressed. The first one considers VQs with memory and focuses specifically on PVQ and FSVQ. The second is concerned with VQs operating over a class of channels exhibiting memory. Design algorithms developed in this dissertation follow the philosophy of GLA, in which the encoder and decoder functions are iteratively improved to each other. Thus, the fundamental problem studied here is the optimal structures for encoder and decoder under a given set of constraints. We have used the mean square error as the distortion measure due to its mathematical tractability and its effectiveness in many relevant applications. The source density function and a statistical description of the channel are assumed known. One of the main objectives of this work has been to use soft-decoding based on continuous-valued channel outputs, which can be viewed as a natural approach to solving the underlying estimation problem. The algorithms proposed in this dissertation do not necessarily yield theoretically optimal codes. However, comparisons with known results are provided to demonstrate that these algorithms do provide good codes. These comparisons are mostly based on the quantization of first-order Gauss-Markov source, described in Appendix C. This source is commonly used as a benchmark for comparing different source coding techniques, as its rate-distortion function can be either computed or bounded [53]. It also provides a good model for real world data in many cases.

Chapter 2 of this dissertation presents basic results in both VQ and COVQ, which will be extended in the following chapters. Some properties of quantizers optimized to noisy channels are also discussed there.

In Chapter 3, the design of PVQ systems based on linear predictors for noisy channels is studied. In particular, an algorithm for designing a PVQ system for noisy channels is derived. This algorithm iteratively adapts a given (usually optimized to noise-free channel) PVQ system, including the predictor, to a given channel. Simulation results presented here show that the proposed algorithm yields codes with performance nearly identical to that of predictive codes obtained by gradient-search optimization techniques in [22]. Furthermore, the proposed algorithm can also be used to design PVQ systems with soft-decoders which provide a further improvement in the overall distortion.

In Chapter 4, the design of FSVQ for noisy channels is studied. The main contribution of the chapter is a robust decoder for a general FSVQ, operating over a noisy channel. In deriving this decoder, we view the operation of an FSVQ as one of choosing a codevector for a given input vector from the set of vectors formed by the union of all state codebooks (*super codebook*). The optimal decoder thus computes the conditional expectation of the super codebook, given the sequence of observed channel outputs. Note that this decoder does not attempt to track the encoder state sequence. We derive a time-recursive algorithm of fixed complexity for computing the output of the optimal decoder. We also develop an iterative algorithm for optimizing an encoder-decoder pair to a given source and a channel. Both hard and soft decoding is considered. Simulation results are obtained for Gauss-Markov process, which show that the proposed FSVQ design methodology yields codes which exhibit graceful degradation of performance with channel noise. Furthermore, robust FSVQs designed here are shown to outperform memoryless COVQ operating at the same rate.

In Chapter 5, the design of COVQ for a class of channels with intersymbol interference is investigated. We focus on soft-decoding, which can be viewed as a generalization of minimum mean square error (MMSE) channel equalization. In particular, we investigate a soft-decoder based on *sliding-block smoothing*, which can be implemented by multi-dimensional function estimation (regression) techniques. The resulting decoder is a non-linear time-invariant MMSE filter, whose parameters may be estimated off-line. In simulation experiments, multi-layer perceptron (MLP) is used to implement the sliding-block decoder. In these simulations quantization of Gauss-Markov source over linear Gaussian channels is considered. According to the results obtained, sliding-block decoder outperforms a probabilistic recursive soft decoder (resembling Kalman filter) studied in [32] at moderate to high channel noise levels. However, a significant computational effort may be required to estimate the sliding block decoder from training data. In particular, the complexity of the optimal mapping appears to increase with the encoder resolution. In this context, we show that, as the channel signal-to-noise ratio is decreased, the optimal sliding-block decoder for a Gaussian channel approximates a linear mapping. This implies that the estimation task becomes simpler as the channel becomes noisier, an observation supported by the experimental results.

In Chapter 6, some directions for future research are outlined.

Chapter 2

Quantizers for Noisy Channels

2.1 Vector Quantization

The basic process of vector quantization (VQ) [1] is illustrated in Fig. 2.1. Let $\mathbf{X} \in \mathbb{R}^d$ be a stationary random vector source, where $\mathbb{R}$ denotes the set of real numbers. The encoder $\epsilon(.)$ is a mapping from d-dimensional real space onto the finite set of integers $\mathbb{I}_N = \{1, 2, \ldots, N\}$, *i.e.*, $\epsilon : \mathbb{R}^d \rightarrow \mathbb{I}_N$. Thus, the encoder partitions $\mathbb{R}^d$ into a set of N non-overlapping cells, denoted by $\{\Omega_1, \Omega_2, \ldots, \Omega_N\}$, and all vectors $\mathbf{X} \in \Omega_i$ are labeled with integer i. We assume that $\cup_{i=1}^{N} \Omega_i = \mathbb{R}^d$. The *rate* R_s of the vector quantizer is defined as $(1/d) log_2 N$ bits per vector component. In this dissertation, we consider only fixed-rate quantization in which the encoder output is to be represented by fixed-length codewords. The output of the encoder is transmitted through a channel to the decoder. In the presence of channel coding, we assume that the channel is noiseless. The decoder $\delta(.)$ is a one-to-one mapping from set $\mathbb{I}_N$ onto the set $\mathbb{C}_N = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_N\} \subset \mathbb{R}^d$, referred to as the *codebook*, *i.e.*, $\delta : \mathbb{I}_N \rightarrow \mathbb{C}_N$. The vectors $\mathbf{c}_i$, $i = 1, \ldots, N$ are called *code vectors* or *reconstruction vectors*. Note

[1] We will use the acronym VQ to refer to both vector quantizer and vector quantization.

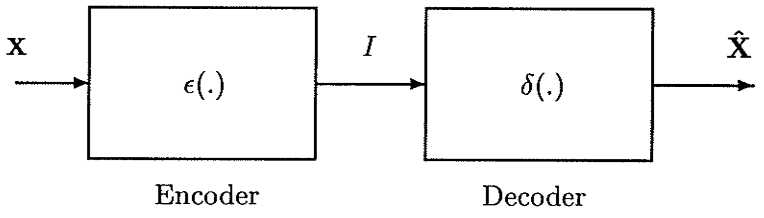


Figure 2.1: *Vector quantization.*

that the decoder is merely a lookup table operation using the encoder output I as an index into the codebook. It is clear that the quantizer approximates all source vectors $\mathbf{X} \in \Omega_i$ by $\mathbf{c}_i$.

The overall operation of an N-level vector quantizer Q_N is the mapping of $\mathbb{R}^d$ onto the finite set $\mathbb{C}_N$, *i.e.*

$$Q_N : \mathbb{R}^d \rightarrow \mathbb{C}_N. \tag{2.1}$$

The distortion of this mapping is measured by an appropriately chosen non-negative cost function $D(\mathbf{X}, \hat{\mathbf{X}})$, which describes the error in approximating the vector $\mathbf{X}$ by $\hat{\mathbf{X}} \in \mathbb{C}_N$. Then, the *globally optimal* VQ for a given source can be defined as

$$(\epsilon^*, \delta^*) = \arg \inf_{\epsilon,\delta} E\{D(\mathbf{X}, \hat{\mathbf{X}})\}, \tag{2.2}$$

where the infimum is taken over all N-level encoder-decoder pairs. A perceptually motivated distortion measure appropriate for many applications, including speech and image coding, is the input-weighted square error [3]

$$D(\mathbf{X}, \hat{\mathbf{X}}) = (\mathbf{X} - \hat{\mathbf{X}})^T W(\mathbf{X})(\mathbf{X} - \hat{\mathbf{X}}), \tag{2.3}$$

where $W(\mathbf{X})$ is a positive definite weighting matrix. The choice of weighting matrix allows one to put more emphasis on certain vector components. When $W(\mathbf{X})$ is the identity matrix, we have the commonly used square error (Euclidean distance) given by

$$D(\mathbf{X}, \hat{\mathbf{X}}) = \|\mathbf{X} - \hat{\mathbf{X}}\|^2. \tag{2.4}$$

We will generally use this distortion measure in most of our work. However, as indicated in Section 2.4, many results derived here readily extend to the more generalized case given by (2.3).

Optimal Vector Quantizer

Let $\mathbf{X}$ be an absolutely continuous vector with probability density function (pdf) $p(\mathbf{x})$. Then, the average distortion of Q_N is given by

$$E\{D(\mathbf{X}, Q_N(\mathbf{X}))\} = \int_{\mathbb{R}^d} \|\mathbf{x} - Q_N(\mathbf{x})\|^2 p(\mathbf{x}) d\mathbf{x}. \tag{2.5}$$

In order to find the globally optimal vector quantizer for a given N and $p(\mathbf{x})$, one has to solve (2.2), a problem one cannot hope to solve analytically in any general sense. In practice, the most widely used approach to designing VQ's is the *generalized Lloyd algorithm* (GLA) [12]. This algorithm is based on solutions to two simpler problems: (i) optimization of the encoder for a fixed decoder, and (ii) optimization of the decoder for a fixed encoder. Solution of these two problems leads to the well known *necessary conditions for optimality* of VQ [3]. In GLA, these two conditions are applied iteratively to the encoder and decoder (starting from some initial choice), until the changes between consecutive iterations are small enough. Since the distortion function in general can have many local minima, this descent algorithm can only locate a local minimum. However, in practice it is known to yield very effective designs. More details on GLA can be found in [12] and Chapter 11 of [3].

Optimal encoder- Given a decoder δ (*i.e.*, a codebook), the optimal encoder ϵ^* is given by the *nearest neighbor condition*

$$\epsilon^*(\mathbf{x}) = i \text{ if } \|\mathbf{x} - \delta(i)\|^2 \leq \|\mathbf{x} - \delta(j)\|^2 \ \forall j \neq i, \tag{2.6}$$

(ties broken arbitrarily) and the optimal encoder cells are the *Voronoi* regions [3] of the code vectors $\delta(i) = \mathbf{c}_i$. These cells are convex polytopes in $\mathbb{R}^d$ [3]. (2.6) can be written in the form

$$\epsilon^*(\mathbf{x}) = i \text{ if } b_i - 2\mathbf{x}^T\mathbf{a}_i \leq b_j - 2\mathbf{x}^T\mathbf{a}_j \ \forall j \neq i, \tag{2.7}$$

where $\mathbf{a}_i = \delta(i)$ and $b_i = \|\delta(i)\|^2$. Thus, we can represent the encoder using a set of parameters $\{(\mathbf{a}_i, b_i),\ i = 1, \ldots, N\}$, which can be computed directly from the given codebook. An important property of the optimal quantizer is that the code vectors are the centroids of the Voronoi partition generated by the code vectors themselves.

Optimal decoder- Given an encoder ϵ (*i.e.*, a partition of $\mathbb{R}^d$), the optimal code vectors are given by

$$\begin{aligned} \delta^*(i) &= \frac{\int_{\Omega_i} \mathbf{x}p(\mathbf{x})d\mathbf{x}}{\int_{\Omega_i} p(\mathbf{x})d\mathbf{x}} \\ &= E\{\mathbf{X}|I = i\}, \end{aligned} \tag{2.8}$$

where $I = i \Leftrightarrow \mathbf{X} \in \Omega_i$. Thus, the optimal code vectors $\delta^*(i) = \mathbf{c}_i^*$ are the centroids of encoder cells Ω_i, $i = 1, \ldots, N$. This condition is known as the *centroid condition.*

The necessary conditions for optimality, given by (2.7) and (2.8), provide the basis for the iterations of GLA. The evaluation of (2.8) requires the knowledge of the input pdf, which in general may not be available. Even if it is known, the analytical evaluation of multi-dimensional integrals over polytopal encoder regions is usually very difficult. Hence, the common practice is to replace the expectations by sample averages based on a set of empirical observations of the input vector *i.e., a training-set.* This obviously requires assumptions on ergodic properties of the source.

In order to start the iterations, an initial encoder $\epsilon^{(0)}$, or equivalently an initial partition of the training set is required. There exists a number of methods for generating the initial partition, a survey of which can be found in [3]. Given the values computed in the $(m-1)^{th}$ iteration, the m^{th} iteration of GLA proceeds as shown in Table 2.1. It is clear that each iteration must either reduce or leave unchanged the average distortion, and hence D_m is a non-increasing sequence. In fact, it can be shown that, under certain mild restrictions, the sequence of quantizers $Q((\epsilon^{(m)}, \delta^{(m)}))$ based on a finite size training set converges to a fixed-point, *i.e.*, one corresponding to a local minimum of the distortion function (*locally optimal quantizer*), see page 356 of [3].

```
1. Compute optimal decoder $\delta^{(m)}$, given $\epsilon^{(m-1)}$ ((2.8)).
2. Compute optimal encoder $\epsilon^{(m)}$, given $\delta^{(m)}$ ( (2.7)).
3. Compute average distortion $D_m = D(\epsilon^{(m)}, \delta^{(m)})$.
4. If $(D_{m-1} - D_m)/D_{m-1}$ is small enough stop.
5. $m \leftarrow m + 1$; Go to 1.
```

Table 2.1: *An iteration of generalized Lloyd algorithm.*

2.2 Vector Quantization for Noisy Channels

We now consider the problem of using vector quantization as a means of source coding (signal compression) for transmitting continuous signals over noisy channels. This situation can be modeled by inserting the channel between the encoder and the

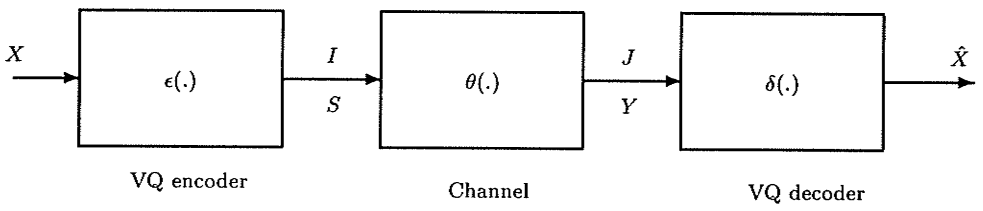


Figure 2.2: *Vector quantization problem for noisy channels.*

decoder as shown in Fig. 2.2. We obtain the VQ shown in Fig. 2.1 if the channel is distortionless or *ideal.* Hence, we refer to the VQ in Fig. 2.1 as the *ideal channel* vector quantizer. Note that there is no explicit channel coding in the system shown in Fig. 2.2. The channel θ in Fig. 2.2 can be modeled in two different ways, leading to two distinct types of decoding strategies. First, if we assume that the channel is discrete and memoryless, $I, J \in \mathbb{I}_N$, and we have $\theta : \mathbb{I}_N \to \mathbb{I}_N$. In this case, the encoder output I is transmitted over the channel, which is received by the decoder as J. Due to random distortion caused by the channel, J may not always be identical to I, and the conditional probability of J, given I is the channel transition probability

$$\Pr(J = j | I = i) = P_{ij}, \quad i, j = 1, \ldots, N. \tag{2.9}$$

The decoder is a one-to-one mapping between the channel output and the reconstruction codebook, *i.e.*, $\delta_H : \mathbb{I}_N \to \mathbb{C}_N$. Such a decoder is referred to as a *hard-decoder.*

Alternatively, one can also consider a continuous channel whose input is an L dimensional vector $\mathbf{S}$ from a given (fixed) set $\mathbb{S}_N = \{\alpha_1, \ \ldots \ , \alpha_N\} \subset \mathbb{R}^L$ and the output $\mathbf{Y}$ is any vector in $\mathbb{R}^L$, *i.e.*, $\theta : \mathbb{S}_N \to \mathbb{R}^L$. The set $\mathbb{S}_N$ is called the *signal constellation*[2]. In this case, the encoder output $I = i$ is mapped to the channel signal

[2]If the channel is K-ary, where $K < N$, K^R-ary channel is obtained by R uses of the channel per source vector.

α_i which is received as a random vector $\mathbf{y}$ by the decoder. The conditional density function of $\mathbf{Y}$ given the channel input vector $\mathbf{S}$ is

$$p(\mathbf{y}|\mathbf{S} = \alpha_i) = p_i(\mathbf{y}), \quad i = 1, \ldots, N. \tag{2.10}$$

As the mapping from $\mathbb{I}_N$ to $\mathbb{S}_N$ is one-to-one, we can now view VQ encoder outputs as being vector valued indices α_i, $i = 1, \ldots, N$. Hence the only difference between the discrete case and the continuous case is in the received "indices" J and $\mathbf{Y}$; J is obtained through a decision on $\mathbf{Y}$. In a communication system, this is achieved by using a *detector* which makes the decision as to which channel input was responsible for the observed output $\mathbf{Y}$. However, when the quantizer minimizes end-to-end distortion (with no channel coding being used), it will be shown below that decoding is an MMSE *estimation* problem. Hence, it is more natural in this case to use the continuous channel output rather than a discretized version of it, as the decoder (estimator) input. A *soft-decoder* is thus a mapping $\delta_S : \mathbb{R}^L \to \mathbb{R}^d$, where $\mathbf{Y} \in \mathbb{R}^L$ is the channel output vector. Clearly, there is a loss of information when the estimation of the source vector in the receiver is based on discretized channel output. The performance improvement due to soft-decoding can be very significant under noisy conditions.

We now turn our attention to the problem of finding the optimal VQ for a given source and a channel. In the remainder of this chapter, we assume that both the quantizer and the channel are memoryless. These restrictions will be relaxed in the following chapters. We further assume that the channel is fixed and its characteristics are completely known. In the case of a discrete channel, we have the discrete memoryless channel (DMC), characterized by its transition probabilities. In the case of soft-decoding, the equivalent characterization is a set of conditional density functions.

A channel of considerable interest is the additive-noise channel described by

$$\mathbf{Y} = \mathbf{S} + \mathbf{W}, \tag{2.11}$$

where $\mathbf{W} \in \mathbb{R}^L$ is the channel noise vector with pdf $p_W(\mathbf{w})$. If the channel noise is independent of signal $\mathbf{S}$, we have

$$p_i(\mathbf{y}) = p_W(\mathbf{y} - \alpha_i). \tag{2.12}$$

Let an N-level encoding function $\epsilon(\mathbf{x})$ be defined by a partition of $\mathbb{R}^d$ into N non-overlapping cells Ω_i, $i = 1, \ldots, N$. Also let $\delta_S(\mathbf{y})$ be a soft decoding function. The distortion caused by the resulting "noisy channel" quantizer Q_N^{NC} is

$$D(\mathbf{X}, Q_N^{NC}(\mathbf{X})) = \|\mathbf{X} - \delta_S(\mathbf{Y})\|^2. \tag{2.13}$$

Mean distortion per vector component is then given by

$$\begin{aligned} E\{\|\mathbf{X} - \delta(\mathbf{Y})\|^2/d\} &= \frac{1}{d}\int_{\mathbb{R}^d}\int_{\mathbb{R}^L} \|\mathbf{x} - \delta_S(\mathbf{y})\|^2 p(\mathbf{x},\mathbf{y}) d\mathbf{x} d\mathbf{y}, \\ &= \frac{1}{d}\sum_{i=1}^{N}\int_{\Omega_i}\int_{\mathbb{R}^L} \|\mathbf{x} - \delta_S(\mathbf{y})\|^2 p_i(\mathbf{y}) p(\mathbf{x}) d\mathbf{y} d\mathbf{x}. \end{aligned} \tag{2.14}$$

where $p(\mathbf{x}, \mathbf{y})$ is the joint density function of $(\mathbf{X}, \mathbf{Y})$. In the case of hard decoding, the channel output $\mathbf{Y}$ has to be simply replaced by the discrete variable $J \in \mathbb{I}_N$ and the average distortion is thus given by

$$E\{\|\mathbf{X} - \delta(\mathbf{J})\|^2/d\} = \frac{1}{d}\sum_{i=1}^{N}\sum_{j=1}^{N} P_{ij}\int_{\Omega_i} \|\mathbf{x} - \delta(j)\|^2 p(\mathbf{x}) d\mathbf{x}. \tag{2.15}$$

The quantizer which minimizes the distortion measures in (2.14) or (2.15) is called a *channel optimized vector quantizer* (COVQ) [40]. Such a quantizer is a function of both source and channel statistics. We next derive the necessary conditions for optimality of the encoder and the decoder for a noisy channel, which can be used to iteratively design COVQs [16], [40], [25].

Optimal Encoder

We can re-write (2.14) as

$$E\{D\} = \frac{1}{d}\sum_{i=1}^{N}\int_{\Omega_i} E\{\|\mathbf{x} - \delta_S(\mathbf{Y})\|^2|\mathbf{S} = \alpha_i\}p(\mathbf{x})d\mathbf{x}. \tag{2.16}$$

Since $p(\mathbf{x})$ is non-negative, for a fixed channel and a decoder, the encoder which minimizes the above function is obtained by choosing a partition such that the conditional expectation $E\{\|\mathbf{x} - \delta(\mathbf{Y})\|^2|\mathbf{S} = \alpha_i\}$ is minimized for every $\mathbf{x}$. Hence, the optimal encoder partition is given by

$$\Omega_i^* = \left\{\mathbf{x} : E\{\| \mathbf{x}-\delta_S(\mathbf{Y}) \|^2 |\mathbf{S} = \alpha_i\} \leq E\{\| \mathbf{x}-\delta_S(\mathbf{Y}) \|^2 |\mathbf{S} = \alpha_j\} \,\forall\, j \neq i\right\}, \tag{2.17}$$

which can be written in the form

$$\Omega_i = \left\{\mathbf{x} : b_i - 2\mathbf{a}_i^T\mathbf{x} \leq b_j - 2\mathbf{a}_j^T\mathbf{x} \,\forall\, j \neq i\right\}, \tag{2.18}$$

where $\mathbf{a}_i = E\{\delta(\mathbf{Y})|\alpha_i\}$ and $b_i = E\{\|\delta(\mathbf{Y})\|^2|\alpha_i\}$ are the encoder parameters, $i = 1, \ldots, N$. Then, the optimal encoder can be described by

$$\epsilon^*(\mathbf{x}) = i \quad \text{if} \quad 2(\mathbf{a}_j - \mathbf{a}_i)^T\mathbf{x} \leq (b_j - b_i) \;\forall j \neq i. \tag{2.19}$$

It can be seen that Ω_i and $\{\Omega_j,\ j = 1,\dots N,\ j \neq i\}$ are separated by a set of hyper planes $2(\mathbf{a}_j - \mathbf{a}_i)^T\mathbf{x} - (b_j - b_i) = 0$ and the encoder partition consists of convex polytopes, as in the case of a quantizer optimized for the ideal channel. Furthermore, by comparing (2.7) and (2.19), it can be seen that the computational complexity of the encoder in the case of noisy channel is identical to that in the case of ideal channel.

The computation of encoder parameters in (2.19) requires the evaluation of L-dimensional integrals. In practice, these expectations are replaced by sample averages based on training sequences. In the case of hard-decoding however, the encoder parameters can be computed directly using the equations

$$\begin{aligned} \mathbf{a}_i &= \sum_{j=1}^{N} \mathbf{C}_j P_{ij}, \\ b_i &= \sum_{j=1}^{N} \|\mathbf{C}_j\|^2 P_{ij}, \quad i = 1,\dots,N, \end{aligned} \tag{2.20}$$

where $\mathbf{C}_j$, $j = 1,\dots,N$ is the decoder codebook.

Optimal Decoder

(2.14) can be re-written as

$$\begin{aligned} E\{D\} &= \frac{1}{d}\int_{\mathbb{R}^L}\int_{\mathbb{R}^d} \|\mathbf{x} - \delta_S(\mathbf{y})\|^2 p(\mathbf{x}|\mathbf{y})d\mathbf{x}d\mathbf{y}, \\ &= \frac{1}{d}\int_{\mathbb{R}^L} E\{\|\mathbf{X} - \delta_S(\mathbf{y})\|^2|\mathbf{y}\}p(\mathbf{y})d\mathbf{y}. \end{aligned} \tag{2.21}$$

The optimal decoder which minimizes $E\{D\}$ is given by the conditional expectation estimator (Appendix A)

$$\delta_S^*(\mathbf{y}) = E\{\mathbf{X}|\mathbf{Y} = \mathbf{y}\}. \tag{2.22}$$

In comparison with (2.8) for ideal channel VQ, we note that optimal decoder in the present case requires the computation of conditional expectation of the encoder input, conditioned on the encoder output observed through the channel. This is called the *regression function* of $\mathbf{X}$ on $\mathbf{Y}$, an interpretation which becomes useful later on. It can be shown that the optimal decoder in (2.22) is the conditional expectation of the encoder centroids, given the channel output, *i.e.*

$$\begin{aligned} E\{\mathbf{X}|\mathbf{Y}=\mathbf{y}\} &= \int_{\mathbb{R}^d} \mathbf{x}p(\mathbf{x}|\mathbf{y})d\mathbf{x} \\ &= \sum_{i=1}^{N} \int_{\Omega_i} \frac{\mathbf{x}p(\mathbf{y}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{y})} d\mathbf{x} \\ &= \frac{\sum_{i=1}^{N} p_i(\mathbf{y}) \int_{\Omega_i} \mathbf{x}p(\mathbf{x})d\mathbf{x}}{\sum_{i=1}^{N} p_i(\mathbf{y}) \int_{\Omega_i} p(\mathbf{x})d\mathbf{x}} \\ &= E\{\mathbb{G}_N|\mathbf{Y}=\mathbf{y}\}, \end{aligned} \tag{2.23}$$

where $\mathbb{G}_N$ is the set of encoder centroids. Note that, in the case of hard-decoding, the optimal decoder is simply

$$\delta^*(j) = \mathbf{C}_j = E\{\mathbb{G}_N|J=j\}, \quad j=1,\ldots,N. \tag{2.24}$$

Here, $\mathbb{C}_N = \{\mathbf{C}_j, j = 1, \ldots, N\}$ is the finite decoder codebook and the decoder is simply a table look-up operation, as in the case of ideal channel VQ. In other words $Q_N^{NC}(\mathbf{X}) \in \mathbb{C}_N$. In contrast, the optimal soft-decoder given by (2.23) implies an infinite codebook, as in that case $Q_N^{NC}(\mathbf{X}) \in \mathbb{R}^d$.

Iterative Design of Noisy Channel VQ

The algorithm given in Table 2.1 can be used to design a COVQ for the given source and channel by replacing (2.7), (2.8), and (2.5) with (2.19), (2.22), and (2.14) (or equivalent results for hard-decoding case) respectively. The alternative application of the optimality conditions in (2.19) and (2.22) cannot increase the average distortion in (2.14), and hence the sequence $\{D_m\}$ (Table 2.1) in this case is clearly non-increasing. As mentioned earlier, the expectations (multi-dimensional integrals) involved in (2.19), (2.22), and (2.14) have to be approximated in practice by sample averages based on training sequences. Although any proof of even local optimality of the resulting quantizers does not exist, the algorithm has shown to yield very effective designs.

Example: COVQ of Gauss-Markov Source over AWGN Channel

The performance of COVQs, designed by GLA for the first-order Gauss-Markov (G-M) source (described in Appendix C) with correlation coefficient 0.9 and additive white Gaussian noise (AWGN) channel is shown in Fig. 2.3. In this example, two-dimensional VQ ($d = 2$) at the rate of 3 bits per vector is considered. That is, if the channel is ideal, the encoder resolution $N = 2^3 = 8$ or the rate $R_s = 2/3$ bits per sample . In order to transmit encoder outputs over the AWGN channel, binary phase shift keying (BPSK) modulation is used. The channel usage rate is fixed at 3 times (bits) per source vector so that, at best, 2/3 bits per sample of source information can be transmitted. In Fig. 2.3, the quality of the channel is indicated by the *channel signal-to-noise ratio* (CSNR). The performance of the VQs are measured by the *signal-to-noise ratio* (SNR), *i.e.*, $E\|\mathbf{X}\|^2/E\|\mathbf{X} - \hat{\mathbf{X}}\|^2$. It can be seen that the system with soft-decoding achieves a performance improvement of around 1 dB in SNR as the channel becomes noisier. Note also that soft-decoding achieves about 2 dB gain in CSNR at SNR of 4 dB. The numbers shown in brackets are the number of cells N in the encoder of the corresponding COVQ, which decrease

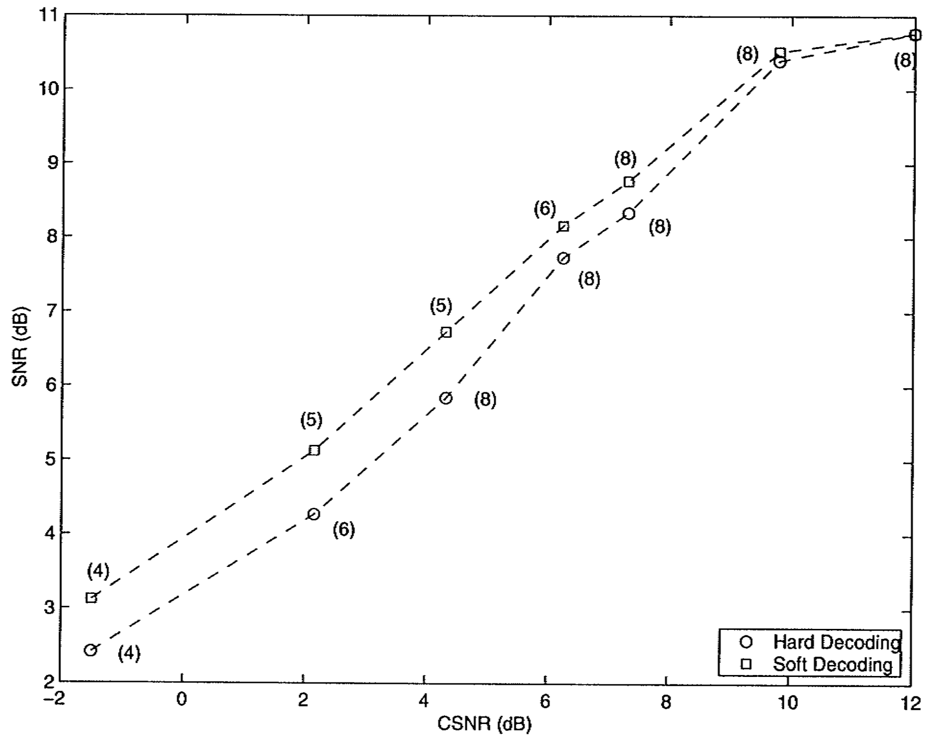


Figure 2.3: *Performance of COVQ for Gauss-Markov source and AWGN channel. Both COVQs were designed using a set of* 100,000 *training vectors. Distortion curves were obtained using a separate set of* 100,000 *vectors.*

as the channel noise level increases. This is due to the fact that COVQ trades-off quantization error for channel error by reducing N below the maximum possible value of 8 in this example. In effect, reducing N increases the average distance between channel signals chosen for transmission. We discuss this further in the next section.

2.3 COVQ and Source/Channel Coding Tradeoff

The overall MSE of a COVQ (with soft-decoding) can be written as (derivation in Appendix B)

$$E\{D\} = \sum_{i=1}^{N} \int_{\Omega_i} \|\mathbf{x} - \mathbf{g}_i\|^2 p(\mathbf{x}) d\mathbf{x} + \sum_{i=1}^{N} \int_{\mathbb{R}^L} \|\mathbf{x} - \delta_S(\mathbf{y})\|^2 p(\mathbf{y}|\alpha_i) P(\alpha_i) d\mathbf{y}, \quad (2.25)$$

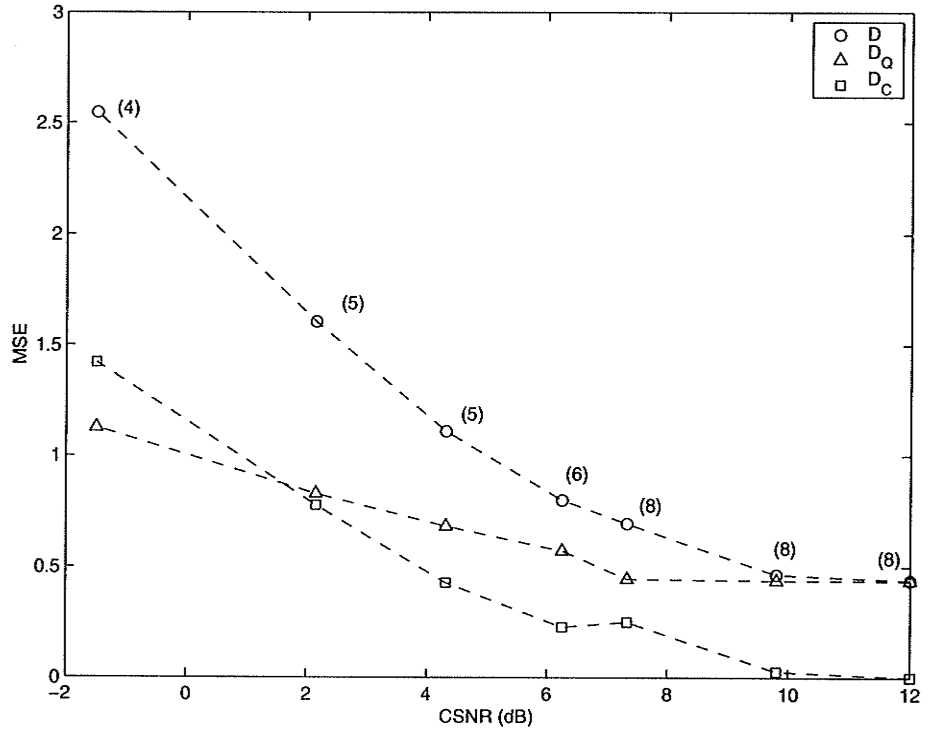


Figure 2.4: *Variation of D, D_Q, and D_c with channel noise level. Variation of D, D_Q, and D_c with channel noise level. The numbers in brackets indicate the number of non-empty encoder cells.*

where $g_i = E\{X|\alpha_i\}$ $i = 1, \ldots, N$ are the encoder centroids. The first sum in the above expression is independent of the channel and is the average distortion when the centroid of each encoding cell is used as the code vector. Hence, it can be viewed as the quantization error D_Q. The second sum in the above expression is the additional distortion due to reproducing a given vector $\mathbf{x}$ by $\delta(\mathbf{y})$ instead of the corresponding encoder centroid $\mathbf{g}_i$, and hence can be considered as the channel error D_C. The variation of D, D_Q, and D_C with channel noise level in the previous example (soft-decoding) is shown in Fig. 2.4. We note that when channel noise level is low, the total error is dominated by the quantization error, while the situation reverses as the channel becomes noisier.

Consider fixed-rate, d-dimensional VQ of a sampled analog source at the rate of R_s

bits per sample. The output of the quantizer is transmitted across a binary channel through n uses of the channel (n-bit channel codewords). Note that no explicit channel coding is used. Define the *transmission rate* R as the number of channel uses per sample, *i.e.*, $R = n/d$. Also define the *channel code rate* $R_c = dR_s/n$ as the number of source bits per channel use so that $R_c = R_s/R$. For a given source sampling rate, R determines the channel uses per second, which is restricted by the available channel bandwidth. Hence an important question is: which quantizer rate R_s minimizes the average distortion for a given R ? Clearly, if the channel is ideal, one can use $R_s = n/d$ bits per sample, or equivalently an encoder with $N = 2^n$ cells. However, if the channel is noisy, some redundancy must be available in channel codewords to protect the transmitted information against channel noise. As we have observed earlier, COVQ trades-off the encoder resolution for increased redundancy in the encoder output.

The *redundancy* in the encoder output of a VQ refers to the amount of extra bits (over the minimum required) used to represent a source sample by the encoder. Let $\{I_n\}$ be the encoder output (index) process. *Entropy* of the encoder index process H_1 is the lowest rate of scalar coding of $\{I_n\}$. Then, the redundancy in I_n due to non-uniform probabilities is given by

$$r_{nu} = dR - H_1. \tag{2.26}$$

Further redundancy may be present when $\{I_n\}$ has memory. Under such conditions, the lowest rate at which $\{I_n\}$ can be coded is given by the *entropy rate* of the process

$$H_\infty = \lim_{l\to\infty} \frac{H(I_1,\ldots,I_l)}{l}, \tag{2.27}$$

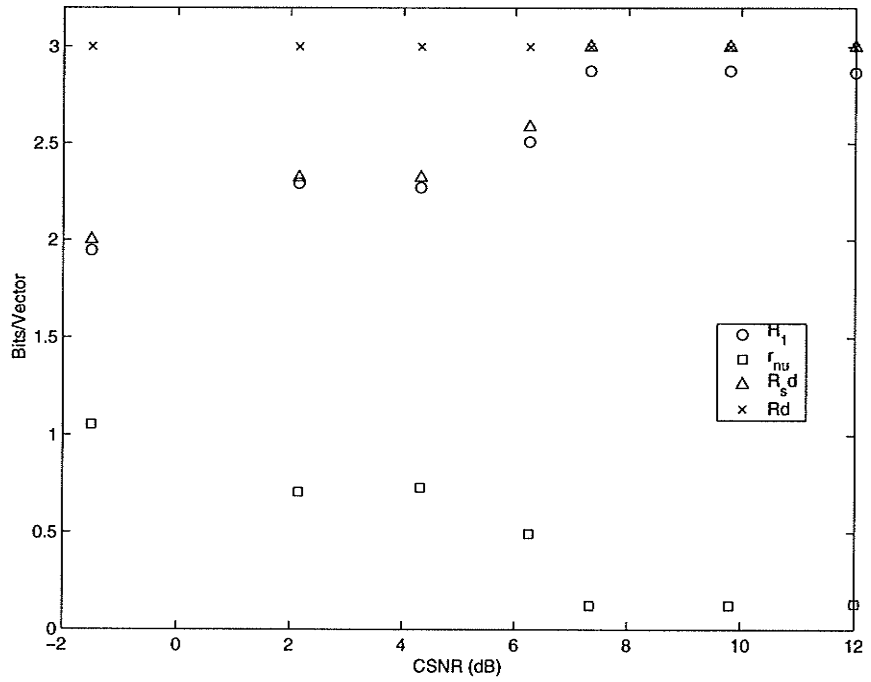


Figure 2.5: *Entropy variation of COVQ with channel noise level.*

where $H(I_1, \ldots, I_l)$ is the joint entropy of a sequence of indices and $H_\infty \leq H_1$. The redundancy due to memory is thus defined as

$$r_m = H_1 - H_\infty. \tag{2.28}$$

If the index process is memoryless, $H_\infty = H_1$ and $r_m = 0$. The total redundancy in the encoder output is given by

$$r_t = r_{nu} + r_m. \tag{2.29}$$

Note that r_t depends on both source statistics and the encoder. Figure 2.5 shows the variation of H_1, R, R_s, and r_{nu} for the soft-decoding COVQ in the previous example. Here, R and R_s are shown in terms of bits per vector to facilitate comparison with other two quantities. Recall that, in this example, R = 1.5 bits per sample. At CSNR =12 dB, the channel is nearly noiseless, and the encoder operates at the rate of $R_s = 1.5$ bits per sample (*i.e.*, N=8). As the channel noise level increases, the entropy of the encoder output H_1 does not appear to change significantly until CSNR drops to about 8 dB. In this range, almost entire transmission rate is used to carry source information and little of it seems to be utilized as protection against channel noise. However, as the channel gets noisier, the source rate decreases, leaving more channel bits for error protection. At -1.5 dB for example, the source rate is only about 2 bits per vector ($N = 4$), leaving almost 1 bit out of 3 channel bits per vector for protection against channel errors. When the channel is noisy, COVQ with fixed R achieves the increase in r_{nu} by reducing the encoder entropy H_1 (see (2.26)), which eventually requires the reduction of the number of encoder cells N and hence the quantizer rate R_s. In essence, a COVQ acts both as a source coder and a channel coder by having the optimal amount of redundancy in the encoder output. At one extreme, when there is no channel noise COVQ is simply a pure source coder. At the other extreme, when the noise variance is infinite, it is the trivial encoder which approximates the input signal by its mean value (*i.e.*, mean distortion is the signal variance). Note that, as the encoder and decoder are memoryless, r_m is not manipulated for any advantage in this example. In Chapters 3, 4, and 5, we study COVQ which employ memory in the encoder and the decoder.

2.4 Generalized Distortion Measure

In this section, we indicate how the optimality conditions given by (2.22) and (2.19) can be extended to the case of input-weighted square error given by (2.3). In this case, the optimal soft-decoder is

$$\delta^*(\mathbf{y}) = \arg\min_{\delta} \int_{\mathbb{R}^d} (\mathbf{x} - \delta(\mathbf{y}))^T W(\mathbf{x})(\mathbf{x} - \delta(\mathbf{y}))d\mathbf{x}$$

$$= \Big[E\{W(\mathbf{X})|\mathbf{y}\}\Big]^{-1}\Big[E\{W(\mathbf{X})\mathbf{X}|\mathbf{y}\}\Big], \tag{2.30}$$

where

$$\begin{aligned} E\{W(\mathbf{X})\mathbf{X}|\mathbf{y}\} &= \int_{\mathbb{R}^d} W(\mathbf{x})\mathbf{x}p(\mathbf{x}|\mathbf{y})d\mathbf{x} \\ &= \frac{\sum_{i=1}^{N} p(\mathbf{y}|\alpha_i)\int_{\Omega_i} W(\mathbf{x})\mathbf{x}p(\mathbf{x})d\mathbf{x}}{\sum_{i=1}^{N} p(\mathbf{y}|\alpha_i)\int_{\Omega_i} p(\mathbf{x})d\mathbf{x}} \end{aligned} \tag{2.31}$$

and

$$E\{W(\mathbf{X})|\mathbf{y}\} = \frac{\sum_{i=1}^{N} p(\mathbf{y}|\alpha_i)\int_{\Omega_i} W(\mathbf{x})p(\mathbf{x})d\mathbf{x}}{\sum_{i=1}^{N} p(\mathbf{y}|\alpha_i)\int_{\Omega_i} p(\mathbf{x})d\mathbf{x}}. \tag{2.32}$$

It is straightforward to show that (proof omitted for brevity) optimal encoder is given by

$$\begin{aligned} \epsilon^*(\mathbf{x}) = \Omega_i \Leftrightarrow \quad & \sum_{l,m}[B_i]_{l,m}[W(\mathbf{x})]_{l,m} - 2\mathbf{x}^T W(\mathbf{x})\mathbf{a}_i \\ & < \sum_{l,m}[B_j]_{l,m}[W(\mathbf{x})]_{l,m} - 2\mathbf{x}^T W(\mathbf{x})\mathbf{a}_j \ \forall j \neq i, \end{aligned} \tag{2.33}$$

where

$$\mathbf{a}_i = E\{\delta(\mathbf{Y})|\alpha_i\},\ i = 1, \ldots, N, \tag{2.34}$$

and B_i is the $d \times d$ conditional covariance matrix

$$B_i = E\{\delta(\mathbf{Y})\delta^T(\mathbf{Y})|\alpha_i\},\ i = 1, \ldots, N. \tag{2.35}$$

Chapter 3

Design of Predictive VQ

3.1 Motivation and Goals

In many practical applications of VQ, the input vectors consist of a set of parameters extracted from a block of consecutive signal samples. For example, in speech coding, it is common to use a set of coefficients representing the short term spectral information of the speech signal [54], [55]. In image coding, blocks of adjacent pixels or their transform coefficients are used as input vectors [56], [57]. As a result, the vector process in many applications often exhibits memory; that is, successive vectors are statistically dependent. The memoryless VQ, in which each vector is quantized independently of others in a sequence, ignores the inter-vector correlation. Theoretically, the performance of memoryless VQ can be made arbitrarily close to the rate-distortion function of the signal source, if the dimension of vectors is allowed to grow (*i.e.*, $d \to \infty$) [8]. In practice, one may achieve near optimal performance by using a sufficiently large dimension. However, doing so may often be impractical as, for a given coding rate R_s (bits per sample), the number of code vectors $N = 2^{dR_s}$, which implies that the computational complexity and storage requirements for full search VQ grows exponentially with d. When the vector dimension is restricted, the quantizers with *memory* can provide increased performance at a given rate and a complexity by exploiting the inter-vector correlation. Such quantizers usually perform encoding and decoding functions recursively, and hence is referred to as *feedback quantizers*. The most common examples are *predictive vector quantizers* (PVQ) [3], [46] and *finite-state vector quantizers* (FSVQ) [49]. A key problem with feedback quantizers is the degradation of performance due to propagation of channel errors in the decoder (receiver). That is, an error in a received codeword can affect a number of succeeding outputs of the decoder. In this chapter, we consider the design of PVQ for noisy channels and propose a new iterative design algorithm. In the next chapter, we focus on the design of FSVQ for noisy channels.

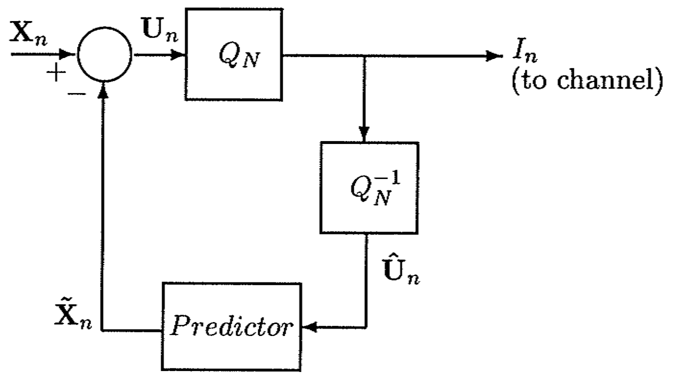
3.2 Predictive Vector Quantization

In this section, a brief introduction to PVQ is provided. A more extensive treatment of the topic may be found in [46], [3]. A block diagram of a PVQ system is shown in Fig. 3.1. Let $\{\mathbf{X}_n\}$ denote a stationary random vector sequence, where $\mathbf{X}_n \in \mathbb{R}^d$. The basic idea of predictive coding is to quantize the error between the actual signal vector at time n, $\mathbf{X}_n$, and its prediction $\tilde{\mathbf{X}}_n$ as shown in Fig 3.1 (a). Note that the prediction is carried-out based on the previous outputs of the quantizer rather than the previous source vectors. This ensures that the predictions made at the encoder and decoder are identical, *i.e.*, $\tilde{\mathbf{X}}'_n = \tilde{\mathbf{X}}_n$. However, this can only be achieved if the channel (not shown) is noiseless. In Fig. 3.1, $\mathbf{U}_n$ and $\hat{\mathbf{U}}_n$ represent the prediction error and its quantized value respectively, and $\hat{\mathbf{X}}_n$ is the reconstructed signal. The encoder Q_N and the decoder Q_N^{-1} used on the prediction error constitute a memoryless, N-level VQ. The prime in the corresponding variables in the decoder indicates the possible difference due to channel noise. Assume that the channel is noiseless, so that the input codeword I_n is identical to the output codeword J_n, and the variables in the encoder and the decoder are identical. The mean square error (MSE) distortion of the PVQ is given by $E\|\mathbf{X}_n - \hat{\mathbf{X}}_n\|^2$. Since

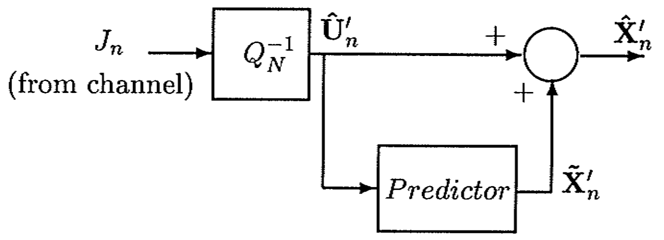
$$\begin{aligned} \mathbf{X}_n - \hat{\mathbf{X}}_n &= \tilde{\mathbf{X}}_n + \mathbf{U}_n - \tilde{\mathbf{X}}_n + \hat{\mathbf{U}}_n \\ &= \mathbf{U}_n - \hat{\mathbf{U}}_n, \end{aligned} \tag{3.1}$$

the overall distortion in PVQ is equal to the distortion in the prediction error signal $\mathbf{U}_n$ due the memoryless quantization by Q_N. If the prediction is good, the error $\mathbf{U}_n$ will be smaller compared to $\mathbf{X}_n$, and for the same quantizer rate, the overall quantization error in PVQ will be much smaller than in ordinary VQ. Alternatively, the same distortion as in ordinary VQ can be achieved in PVQ, using a lower quantizer rate. Note that the predictive encoder is a time varying partition of $\mathbb{R}^d$ and the decoder is a time varying codebook. However, the memoryless encoder/decoder pair (Q_N, Q_N^{-1}) and the predictor are time-invariant. If the encoder Q_N uses the nearest neighbor rule, the predictive encoder is a Voronoi partition with respect to the set of vectors $\{\tilde{\mathbf{x}}_n + \mathbf{c}_i, i = 1, \ldots, N\}$, where $\mathbf{c}_i$ are fixed code vectors.

The most common form of prediction used in PVQ is linear prediction. Non-linear predictors are difficult to design and *ad hoc* designs are often not effective [3]. An exception would be when data is Gaussian, in which case the optimal non-linear predictor is linear. In this thesis, we will exclusively focus on PVQ systems based on linear prediction. In this case, the predictor in Fig. 3.1 is a linear filter which can be



(a) encoder



(b) decoder

Figure 3.1: *Predictive vector quantizer (Q_N and Q_N^{-1} are encoder and decoder respectively of a zero-memory quantizer).*

of either *auto-regressive* (AR) type, given by

$$\tilde{\mathbf{x}}_n = \sum_{k=1}^{P} A_k \hat{\mathbf{x}}_{n-k} \tag{3.2}$$

$$= \sum_{k=1}^{\infty} B_k \hat{\mathbf{u}}_{n-k}, \tag{3.3}$$

or *moving average* (MA) type given by

$$\tilde{\mathbf{x}}_n = \sum_{k=1}^{P'} A'_k \hat{\mathbf{u}}_{n-k}, \tag{3.4}$$

where A_k and A'_k are matrices of prediction coefficients [3]. B_k are matrices which depend on predictor matrices A_k. For example, for first-order prediction $P = 1$ and $B_k = A_1^k$. In practice, it has been observed that systems using AR predictors (AR-PVQ) perform better than those using MA predictors (MA-PVQ) [58]. However, in the presence of channel noise, the performance of AR-PVQ degrades more severely than MA-PVQ, as a channel error propagates over the entire sequence of decoder outputs following the error. In order to see this, let $\mathbf{c}_n$ be the channel error in the decoded value of the prediction error $\hat{\mathbf{u}}'_n$ at the receiver. Then, from (3.3) the output of an AR-PVQ system at time $m \geq n$ is

$$\hat{\mathbf{x}}'_m = \hat{\mathbf{x}}_m + B_{m-n}\mathbf{c}_n,$$

where the second term on right-hand side is the propagated channel error. On the other hand, the error propagation problem in a MA-PVQ systems is limited to only P' vectors following the error, see (3.4). A comparison of noisy channel performance between AR-PVQ and MA-PVQ for speech coding can be found in [45]. In this chapter, we will focus on AR-PVQ systems. However, the derivations presented in the following sections are also applicable to MA-PVQ.

A block diagram of an AR-PVQ system is shown in Fig. 3.2. In the absence of channel noise, the error decoder δ_1 and the predictor β_1 on encoder side are identical to their counterparts δ_2 and β_2 respectively on the decoder side. Due to feedback structure of the encoder and decoder, the analysis of PVQ is more difficult than the analysis of memoryless VQ and there exists no known design algorithm which guarantees even a locally optimal quantizer. However, several heuristic based design algorithms which result in "good" PVQs do exist, and the designs obtained with these algorithms convincingly outperform simple memoryless VQ, when the source has memory. An overview of different approaches may be found in Chapter 13 of

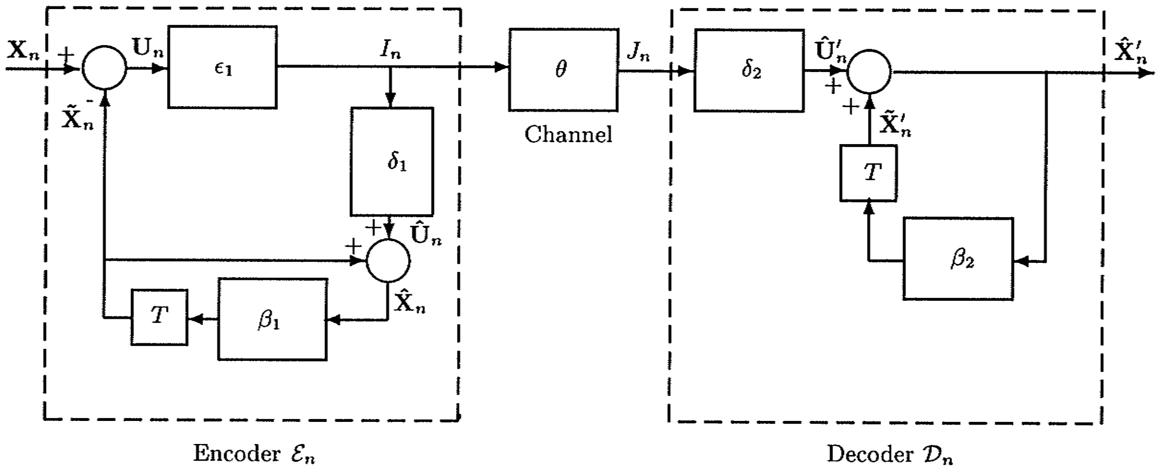


Figure 3.2: *PVQ with an AR predictor.*

[3]. The most effective design algorithm is the Lloyd-style algorithm introduced in [46]. In this algorithm, an open-loop predictor is first designed for the unquantized input vector source, using optimal linear prediction theory. This predictor is then used in the closed-loop PVQ system and is not changed further. The encoder ϵ_1 and the decoder δ_1 in the feedback loop are then improved using Lloyd iterations for memoryless VQ. In order to avoid difficulties due to feedback, each Lloyd iteration is carried out for a fixed prediction error sequence which is then updated for the next iteration. Clearly, this procedure does not result in a locally optimal codebook. Nonetheless, experimental results show that very effective designs can be obtained using this procedure. An alternative PVQ design method is the gradient search optimization procedure investigated in [59]. In this approach, both codebook and the predictor coefficients are updated simultaneously, using gradient descent methods. Despite being a theoretically better approach, it has been found that designs obtained with this algorithm perform almost identically to those obtained with the simpler Lloyd-style algorithm in [46]. In this chapter, we will confirm that this observation holds true even in the case of noisy channel designs.

3.3 Noisy Channel PVQ Problem

Consider the AR-PVQ system shown in Fig. 3.2. In the presence of channel noise, the channel input I_n and the channel output J_n are not always identical. Assume for the time being that the channel θ is a discrete memoryless channel (DMC) with transition probabilities $Pr(J_n = j|I_n = i) = p_{ij}$, where $I_n, J_n \in \mathbb{I}_N = \{1, 2, \ldots, N\}$. Assume also that the channel is fixed and known. Let the predictive encoder and decoder be denoted by $\mathcal{E}_n(\epsilon_1, \delta_1, \beta_1)$ and $\mathcal{D}_n(\delta_2, \beta_2)$ respectively. We wish to find the system components $\{\epsilon_1, \delta_1, \delta_2, \beta_1, \beta_2\}$ which minimize the average distortion

$$\begin{aligned} E\{D_n\} &= E\|\mathbf{X}_n - \hat{\mathbf{X}}'_n\|^2 \\ &= \int_{\mathbb{R}^d}\int_{\mathbb{R}^d} \|\mathbf{x}_n - \hat{\mathbf{x}}'_n\|^2 p(\mathbf{x}_n, \hat{\mathbf{x}}'_n) d\mathbf{x}_n d\hat{\mathbf{x}}'_n. \end{aligned} \tag{3.5}$$

It should be noted that, even though $\{\mathbf{X}_n\}$ is stationary, the system in general is not. A recursive system, which is initialized at some point in time may not be stationary. However, if the encoder and decoder are stable, the system is asymptotically stationary in that, the effects due to initial conditions die-out as $n \to \infty$. Under these conditions the predictor output can be given by (3.3). Furthermore, the expectations can be approximated by long term sample averages.

Our goal in this chapter is to derive an iterative algorithm for designing noisy channel PVQ. As stated earlier, the task of finding the exact necessary conditions for optimality of the encoder and decoder in the case of recursive quantizers is a difficult problem, to which no known solution exits. Hence, we rather focus on finding a solution based on simplifying assumptions, which hopefully leads to a locally optimal system. It should be emphasized that the "optimality conditions" derived here are only approximate and do not necessarily hold true for a closed-loop system. However, the experimental results show that these conditions lead to good codes for noisy channels. Even though the predictors in the encoder and decoder need not necessarily be identical in the presence of channel noise, in order to make the problem tractable, we use the same predictor β in both encoder and decoder. Furthermore, the predictor is assumed to be a linear filter, which includes both AR and MA cases. The derivations presented in the following sections consider the AR predictor given by (3.3). However, the modifications required for the MA predictor are trivial. In the following, we formulate and attempt to solve the following three problems: (1) for a fixed predictor β and a encoder $\mathcal{E}_n$, find optimal decoder $\mathcal{D}_n$, (2) for a fixed predictor β and a decoder $\mathcal{D}_n$ find the optimal encoder $\mathcal{E}_n$, and (3) for a fixed system find the optimal predictor β. We will then present an iterative design algorithm based on these conditions. The algorithm is first derived for the case of discrete memoryless channel (DMC) and hard-decoding. The extension to soft-decoding is straight forward and is considered in Sec. 3.8.

3.4 Optimal Decoder

According to the above formulation, the problem of finding the optimal decoder $\mathcal{D}_n$ for a given encoder $\mathcal{E}_n$ is equivalent to finding the decoder δ_2 which minimizes distortion measure in (3.5), given ϵ_1, δ_1, and β. The output of the decoder $\mathcal{D}_n$ is given by

$$\hat{\mathbf{x}}'_n = \hat{\mathbf{u}}'_n + \tilde{\mathbf{x}}'_n, \tag{3.6}$$

where $\hat{\mathbf{u}}'_n = \delta_2(j_n)$ From (3.3), the AR predictor output can be written as

$$\tilde{\mathbf{x}}'_n = \sum_{k=1}^{\infty} B_k \hat{\mathbf{u}}'_{n-k}. \tag{3.7}$$

Note also that

$$\begin{aligned} \mathbf{x}_n &= \mathbf{u}_n + \tilde{\mathbf{x}}_n \\ &= \mathbf{u}_n + \sum_{k=1}^{\infty} B_k \hat{\mathbf{u}}_{n-k}. \end{aligned} \tag{3.8}$$

Hence, the average distortion in (3.5) can be written as

$$\begin{aligned} D &= E\{\|\mathbf{X}_n - \hat{\mathbf{U}}'_n - \sum_{k=1}^{\infty} B_k \hat{\mathbf{U}}'_{n-k}\|^2\} \\ &= E\{\|\mathbf{U}_n - \hat{\mathbf{U}}'_n + \sum_{k=1}^{\infty} B_k (\hat{\mathbf{U}}_{n-k} - \hat{\mathbf{U}}'_{n-k})\|^2\} \\ &= E\{\|\mathbf{U}_n - \hat{\mathbf{U}}'_n\|^2\} + 2\sum_{k=1}^{\infty} E\{(\mathbf{U}_n - \hat{\mathbf{U}}'_n)^T B_k (\hat{\mathbf{U}}_{n-k} - \hat{\mathbf{U}}'_{n-k})\} \\ &\quad + E\{\|\sum_{k=1}^{\infty} B_k (\hat{\mathbf{U}}_{n-k} - \hat{\mathbf{U}}'_{n-k})\|^2\} \end{aligned} \tag{3.9}$$

In order to proceed further, we now assume that prediction error $\{\mathbf{U}_n\}$ is an uncorrelated process, which is reasonable if the linear predictor is a good predictor. Note that the prediction error of an optimal linear predictor is uncorrelated, see Theorem 4.91 (pp. 119) in [3]. The assumption implies that the second term in the last equation is zero. We further assume that the quantization error $\mathbf{u}_n - \hat{\mathbf{u}}_n$ is small ($\approx \mathbf{0}$). This assumption is a good one for high-rate quantization, but can be poor at low-rates. Based on these assumptions, we obtain an approximate distortion measure, which is a function of only the decoder δ_2.

$$\begin{aligned} E\{D\} &\approx E\{\|\mathbf{U}_n - \hat{\mathbf{U}}'_n\|^2\} + \sum_{k=1}^{\infty} E\{\|B_k(\mathbf{U}_{n-k} - \hat{\mathbf{U}}'_{n-k})\|^2\} \\ &= \sum_{k=0}^{\infty} E\|B_k(\mathbf{U}_{n-k} - \hat{\mathbf{U}}'_{n-k})\|^2, \end{aligned} \tag{3.10}$$

where B_0 is the identity matrix. Since the channel is assumed memoryless, $\{\hat{\mathbf{U}}'_n\}$ is also uncorrelated and the above expression is minimized by minimizing $E\|B_k(\mathbf{U}_{n-k} - \hat{\mathbf{u}}'_{n-k})\|^2$ with respect to $\hat{\mathbf{u}}'_{n-k}$ for every $k \geq 0$. Hence, the optimal decoder $\hat{\mathbf{u}}'^*_n = \delta^*_2(j_n)$ can be found by solving

$$\delta^*_2(j_n) = \arg\min_{\delta_2} E\|B_k(\mathrm{U}_n - \delta_2(j_n))\|^2. \tag{3.11}$$

From (A.5), the solution to this memoryless COVQ problem is

$$\delta^*_2(j_n) = E\{\mathbf{U}_n | J_n = j_n\}. \tag{3.12}$$

That is, we have to find the channel optimized decoder for a given error encoder ϵ_1. This means that, under our assumptions, the optimal PVQ decoder $\mathcal{D}_n$ (with a fixed predictor β) for a given encoder $\mathcal{E}_n$ is obtained by optimally decoding the prediction error. The decoder in (3.12) is given by the codebook $\mathbb{C}^{(2)}_N = \{\mathbf{c}_2(1), \ldots \mathbf{c}_2(N)\}$ (see (2.24)).

$$\mathbf{c}_2(j_n) = \frac{\sum_{i=1}^N \mathbf{g}_i p_{ij_n} P_i}{\sum_{i=1}^N p_{ij_n} P_i}, \tag{3.13}$$

where $\mathbf{g}_i$ is the centroids of the i^{th} encoding cells in ϵ_1, P_i is the prior probability of the i^{th} channel input, and p_{ij} is the channel transition probability defined earlier, $i, j = 1, \ldots, N$.

3.5 Optimal Encoder

In this case, the problem is equivalent to finding optimal ϵ_1 and δ_1, given $\mathcal{D}_n$ and β. Due to the feedback structure in the encoder, there is no obvious way of finding exactly the desired optimality conditions. However, in an iterative algorithm, the optimization of $\mathcal{E}_n$ may be split into two steps: (i) given $\{I_n\}$ and the decoder $\mathcal{D}_n$, find optimal δ_1, (ii) given $\{\mathbf{U}_n\}$ and the decoder $\mathcal{D}_n$, find optimal ϵ_1.

Step (i): Feedback Path

Note that the purpose of the feedback path is to mimic the given combination of channel θ and the decoder (receiver) $\mathcal{D}_n(\delta_2, \beta)$. Ideally, we must place an exact copy of the channel in the feedback path in the encoder $\mathcal{E}_n$. This is clearly impractical and it is desired that the signal reconstructed by δ_1 in the feedback path $\hat{\mathbf{X}}_n$ be as 'close' as possible to the signal $\hat{\mathbf{X}}'_n$ reconstructed at the receiver. Since

$$\hat{\mathbf{x}}_n = \sum_{k=0}^{\infty} B_k \hat{\mathbf{u}}_{n-k}, \tag{3.14}$$

$\hat{\mathbf{X}}_n$ is a function of $\bar{I}_n \triangleq (I_n, I_{n-1}, \ldots, I_{-\infty})$, and in the present problem it will be determined by the choice of δ_1. Therefore, the local decoder δ_1 is designed to minimize the distortion measure

$$E\{\|\hat{\mathbf{X}}_n - \hat{\mathbf{X}}'_n\|^2 | \bar{\imath}_n\}. \tag{3.15}$$

Clearly, if the channel is noiseless, this criterion gives $\delta_1{=}\delta_2$. Using (3.2), the above distortion measure can be expanded as

$$\begin{aligned} E\{\|\hat{\mathbf{X}}_n - \hat{\mathbf{X}}'_n\|^2 | \bar{\imath}_{n-1}\} &= E\{\|\hat{\mathbf{u}}_n + \sum_{k=1}^{\infty} B_k \hat{\mathbf{u}}_{n-k} - \hat{\mathbf{U}}'_n - \sum_{k=1}^{\infty} B_k \hat{\mathbf{U}}'_{n-k}\|^2 | \bar{\imath}_n\} \\ &= E\{\| \sum_{k=0}^{\infty} B_k (\hat{\mathbf{u}}_{n-k} - \hat{\mathbf{U}}'_{n-k})\|^2 | \bar{\imath}_n\} \\ &= E\{\| \sum_{k=0}^{\infty} B_k (\delta_1(\imath_{n-k}) - \hat{\mathbf{U}}'_{n-k})\|^2 | \bar{\imath}_n\}, \end{aligned} \tag{3.16}$$

where $\hat{\mathbf{u}}_{n-k} = \delta_1(i_{n-k})$. If the channel is memoryless, the last equation becomes

$$E\{\|\hat{\mathbf{X}}_n - \hat{\mathbf{X}}'_n\|^2|\bar{i}_n\} = \sum_{k=0}^{\infty} E\{\|B_k(\delta_1(i_{n-k}) - \hat{\mathbf{U}}'_{n-k})\|^2|i_{n-k}\}. \tag{3.17}$$

The optimal decoder $\delta_1^*(i_n)$ is therefore given by (see (A.5))

$$\begin{aligned}\delta_1^*(i_n) &= \arg\min_{\delta_1} E\{\|\delta_1(i_n) - \hat{\mathbf{U}}'_n\|^2|I_n = i_n\} \\ &= E\{\hat{\mathbf{U}}'_n|I_n = i_n\}.\end{aligned} \tag{3.18}$$

This is a codebook $\mathbb{C}_N^{(1)} = \{\mathbf{c}_1(1), \ldots \mathbf{c}_1(N)\}$ with code vectors (see (2.24))

$$\mathbf{c}_1(i) = \sum_{j=1}^{N} \mathbf{c}_2(j)p_{ij}, \quad i = 1, \ldots, N, \tag{3.19}$$

where $\mathbf{c}_2(j)$, $j = 1, \ldots, N$ is the codebook of given decoder δ_2. The above result gives the optimal local decoder for a given channel input sequence. The predicted sequence generated by this choice of δ_1 is given by

$$\begin{aligned}\tilde{\mathbf{x}}_n &= \sum_{k=1}^{\infty} B_k \hat{\mathbf{u}}_{n-k} \\ &= \sum_{k=1}^{\infty} B_k E\{\hat{\mathbf{U}}'_{n-k}|i_{n-k}\} \\ &= E\{\sum_{k=1}^{\infty} B_k \hat{\mathbf{U}}'_{n-k}|i_{n-k}\} \\ &= E\{\tilde{\mathbf{X}}'_n|\bar{i}_{n-1}\}.\end{aligned} \tag{3.20}$$

Step (ii): Feedforward Path

We now consider the feedforward path in the encoder for a fixed feedback path, assuming that the predicted value $\tilde{\mathbf{x}}_n$ is given by (3.20). Given the knowledge of past outputs $\bar{\imath}_{n-1}$ (via the prediction $\tilde{\mathbf{x}}_n$), the optimal $\mathcal{E}_n$ at time n is a partition of $\mathbb{R}^d$ such that

$$i_n^* = \arg\min_i E\{\|\mathbf{x}_n - \hat{\mathbf{X}}'_n\|^2 | I_n = i, \bar{\imath}_{n-1}\}. \tag{3.21}$$

Define the distortion measure

$$D(\mathbf{x}_n | i, \bar{\imath}_{n-1}) = E\{\|\mathbf{x}_n - \hat{\mathbf{X}}'_n\|^2 | I_n = i, \bar{\imath}_{n-1}\}. \tag{3.22}$$

Then, the optimal $\mathcal{E}_n$ is described by

$$i_n^* = i \;\text{ if }\; D(\mathbf{x}_n | i, \bar{\imath}_{n-1}) \leq D(\mathbf{x}_n | l, \bar{\imath}_{n-1}) \;\; \forall\, l \neq i. \tag{3.23}$$

Consider

$$\begin{aligned} D(\mathbf{x}_n | i, \bar{\imath}_{n-1}) &= E\{\|\tilde{\mathbf{x}}_n + \mathbf{u}_n - \tilde{\mathbf{X}}'_n - \hat{\mathbf{U}}'_n\|^2 | I_n = i, \bar{\imath}_{n-1}\} \\ &= E\{\|\mathbf{u}_n - \hat{\mathbf{U}}'_n\|^2 | I_n = i\} + E\{\|\tilde{\mathbf{x}}_n - \tilde{\mathbf{X}}'_n\|^2 | \bar{\imath}_{n-1}\} \\ &\quad +2E\{(\mathbf{u}_n - \hat{\mathbf{U}}'_n)^T (\tilde{\mathbf{x}}_n - \tilde{\mathbf{X}}'_n) | I_n = i, \bar{\imath}_{n-1}\} \\ &= E\{\|\mathbf{u}_n - \hat{\mathbf{U}}'_n\|^2 | I_n = i\} + E\{\|\tilde{\mathbf{x}}_n - \tilde{\mathbf{X}}'_n\|^2 | \bar{\imath}_{n-1}\} \\ &\quad +2E\{(\mathbf{u}_n - \hat{\mathbf{U}}'_n)^T | I_n = i\} (\tilde{\mathbf{x}}_n - E\{\tilde{\mathbf{X}}'_n | \bar{\imath}_{n-1}\}), \end{aligned} \tag{3.24}$$

where we use the fact that $\tilde{\mathbf{X}}_n$ and $\tilde{\mathbf{X}}'_n$ are conditionally independent of I_n. According to (3.20), the last term in the above expression is zero. Hence the inequality in (3.23) simplifies to

$$i_n^* = i \text{ if } E\{\|\mathbf{u}_n - \hat{\mathbf{U}}_n'\|^2 | i_n = i\} \leq E\{\|\mathbf{u}_n - \hat{\mathbf{U}}_n'\|^2 | i_n = l\} \ \forall\, l \neq i, \tag{3.25}$$

which is a function of only ϵ_1. That is, the optimal partition associated with $\mathcal{E}_n$ is obtained by optimally encoding the prediction error using ϵ_1^*, *i.e.*, the channel optimized encoder for the prediction error $\mathbf{U}_n$. Given a prediction error sequence, ϵ_1^* can be found as in Sec. 2.2, and the encoder ϵ_1^* can be described by the set of parameters

$$\begin{aligned} \mathbf{a}_i &= \sum_{j=1}^{N} \mathbf{c}_2(j) p_{ij} \\ b_i &= \sum_{j=1}^{N} \|\mathbf{c}_2(j)\|^2 p_{ij} \end{aligned} \tag{3.26}$$

It is worth noting that, if the channel is noiseless ($J_n = I_n$), the conditions for both ϵ_n^* and δ_n^* reduces to those used in [59] for noiseless channel PVQ design.

3.6 Optimization of Predictor

In this section, we develop a procedure for updating the linear coefficient matrices A_k, $k = 1, \ldots, P$ in a given PVQ system, so that the distortion measure $E\{D\}$ in (3.5) is decreased. The output of the PVQ decoder can be expressed as

$$\hat{\mathbf{x}}_n' = \hat{\mathbf{u}}_n' + \sum_{k=1}^{P} A_k \hat{\mathbf{x}}_{n-k}'. \tag{3.27}$$

Substituting this expression in (3.5) we get

$$\begin{aligned} E\{D\} &= E\|\mathbf{X}_n - \hat{\mathbf{U}}'_n - \sum_{k=1}^{P} A_k \hat{\mathbf{X}}'_{n-k}\|^2 \\ &= E\|\hat{\mathbf{X}}''_n - \sum_{k=1}^{P} A_k \hat{\mathbf{X}}'_{n-k}\|^2, \end{aligned} \tag{3.28}$$

where

$$\hat{\mathbf{X}}''_n = \mathbf{X}_n - \hat{\mathbf{U}}'_n. \tag{3.29}$$

In this equation, the sequence $\{\hat{\mathbf{X}}'_n\}$ can be considered as the input to a prediction filter whose desired output sequence is $\{\hat{\mathbf{X}}''_n\}$. If a data set of sample input and output sequences is available, the optimal filter coefficients $\hat{A}^*_k$ which minimize $E\{D\}$ can be estimated. We proceed in this direction.

Let

$$\underline{\hat{\mathbf{X}}}'_{n-1} = (\hat{\mathbf{X}}'^T_{n-1}, \hat{\mathbf{X}}'^T_{n-2}, \ldots, \hat{\mathbf{X}}'^T_{n-P})^T \tag{3.30}$$

be the dP-dimensional vector formed by concatenating P d-dimensional vectors and

$$A = (A_1,\ A_2\ \ldots,\ A_P) \tag{3.31}$$

be a $d \times dP$ matrix formed by P $d \times d$-dimensional predictor matrices. With this notation, (3.28) becomes

$$E\{D\} = E\|\hat{\mathbf{X}}''_n - A\underline{\hat{\mathbf{X}}}'_{n-1}\|^2. \tag{3.32}$$

In order to find A which minimizes $E\{D\}$, we let $\nabla_A E\{D\} = \mathbf{0}$ so that

$$E\{\hat{\mathbf{X}}''_n \underline{\hat{\mathbf{X}}}'^T_{n-1}\} - AE\{\underline{\hat{\mathbf{X}}}'_{n-1}\underline{\hat{\mathbf{X}}}'^T_{n-1}\} = \mathbf{0}. \tag{3.33}$$

This may be written as

$$\bar{P} - A\bar{R} = \mathbf{0}, \tag{3.34}$$

where $\bar{P} = E\{\hat{\mathbf{X}}_n''\underline{\hat{\mathbf{X}}}_{n-1}'^T\}$ and $\bar{R} = E\{\underline{\hat{\mathbf{X}}}_{n-1}'\underline{\hat{\mathbf{X}}}_{n-1}'^T\}$ are respectively $d \times dP$ and $dP \times dP$ dimensional matrices, which depend on both source and channel statistics. Since $\bar{R}$ is the covariance matrix of the random vector $\underline{\hat{\mathbf{X}}}_n'$, it is positive semi-definite and positive definite if $\{\hat{\mathbf{X}}_n'\}$ is nondeterministic, see Chapter 13 of [3]. In that case, a unique solution to the above matrix form *Wiener-Hopf* equation exists, which can be obtained by

$$A^* = \bar{P}\bar{R}^{-1}. \tag{3.35}$$

In practice, Hermitian positive definite nature of covariance matrices can be exploited for numerically stable and computationally efficient solution of this equation [60], [61]. In this chapter, simulation studies were confined to first-order predictors and hence (3.35) was solved directly. Given a training set of source vectors $\{\mathbf{x}_n\}_{n=1}^{n_T}$, realizations of $\{\underline{\hat{\mathbf{x}}}_n'^{(m)}\}_{n=1}^{n_M}$ and $\{\hat{\mathbf{x}}_n''^{(m)}\}_{n=1}^{n_M}$, $m = 1, \ldots, n_M$ are obtained by transmitting (by simulating the channel) the training set n_M times using the given PVQ system. Then, the required estimates can be obtained as

$$\hat{\bar{P}} = \sum_{n=P+1}^{n_T} \sum_{m=1}^{n_M} \hat{\mathbf{x}}_n''^{(m)} \underline{\hat{\mathbf{x}}}_{n-1}'^{(m)T}, \tag{3.36}$$

$$\hat{\bar{R}} = \sum_{n=P}^{n_T} \sum_{m=1}^{n_M} \underline{\hat{\mathbf{x}}}_n'^{(m)} \underline{\hat{\mathbf{x}}}_n'^{(m)T}. \tag{3.37}$$

After the solution to (3.35) is obtained, the predictor matrices A_k, $k = 1, \ldots, P$ can be updated in the recursive algorithm presented below.

3.7 Design Algorithm

In this section, we present an algorithm for designing noisy channel PVQ using a training set of input vectors. The basic approach used here is inspired by the Lloyd algorithm, and in particular the PVQ design algorithm of [46]. Briefly stated, ϵ_1, δ_1, δ_2, and β are iteratively updated using the conditions derived above, until the average distortion decreases below a specified threshold. It is not necessary that the repeated application of these conditions result in a monotone decrease of distortion. Hence, the convergence of the proposed algorithm remains an open question. However, as in the case of other PVQ design algorithms [46], [59], [22], in practice the present algorithm has shown to yield good codes which were considerably superior to memoryless VQ. We also found that PVQ systems obtained with this algorithm performed comparably with those obtained by the gradient search algorithm of [22].

Given a training set of source vectors $\{\mathbf{x}_n\}_{n=1}^{n_T}$, a good PVQ system can be designed for a noise free channel using, for example, the method of [59]. This system can be used as the initial system which is to be iteratively improved for the given channel. The average distortion is computed using the sample average

$$\bar{D}^{(k)} = \frac{1}{n_T} \sum_{n=1}^{n_T} \frac{1}{n_M} \sum_{j=1}^{n_M} \|\mathbf{x}_n - \hat{\mathbf{x}}_n'^{(k)}(j)\|^2, \tag{3.38}$$

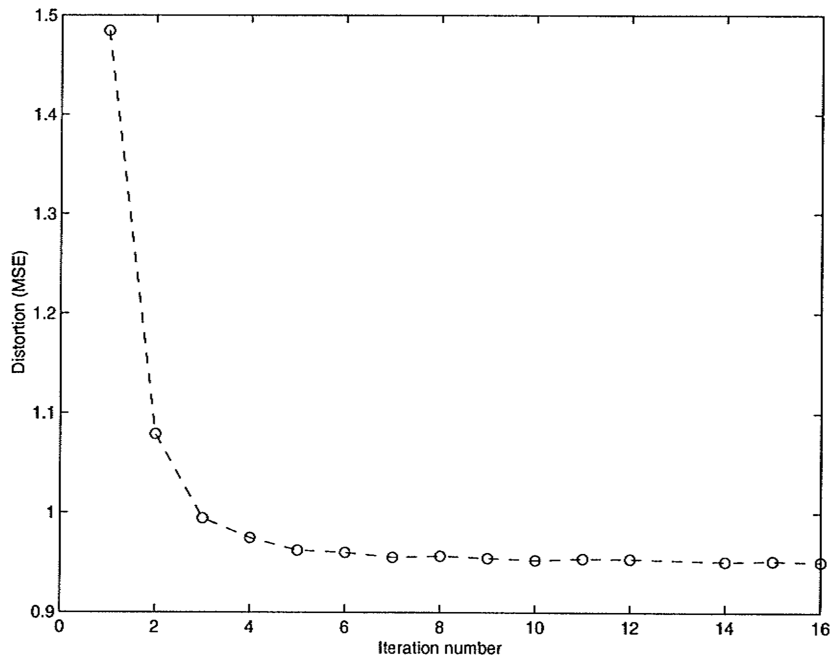
where $\hat{\mathbf{x}}_n'(j)$ is the decoder output when $\mathbf{x}_n$ is encoded and transmitted for the j^{th} time and n_M is the number of times the sequence is transmitted (to average over channel noise). The complete design algorithm is given in Table 3.1.

An important issue in the implementation of this algorithm is the stopping criteria. As mentioned earlier, the algorithm, when applied to a closed-loop system, is not guaranteed to converge. It was noted in all our simulations that the distortion

```
Step 0:  Given: $\epsilon^{(0)}$, $\delta_1^{(0)}$, $\delta_2^{(0)}$, and $\beta^{(0)}$.
         $k \leftarrow 0$.
Step 1:  Compute prediction error sequence $\{\mathbf{u}_n^{(k)}\}$ and channel input
         sequence $\{i_n^{(k)}\}$.
Step 2:  Compute average distortion $\bar{D}^{(k)}$ using (3.38).
Step 3:  If $\bar{D}^{(k)}$ is small enough (convergence-criteria met) stop;
         Else $k \leftarrow k+1$.
Step 4:  Find $\delta_2^{(k)}$ using (3.13).
Step 5:  Find $\delta_1^{(k)}$ using (3.19).
Step 6:  Find $\epsilon_1^{(k)}$ using (3.26).
Step 7:  Update $\{\mathbf{u}_n^{(k)}\}$ and $\{i_n^{(k)}\}$.
Step 8:  Find $\beta^{(k)}$ using (3.35).
         Repeat from Step 1.
```

Table 3.1: *Proposed noisy channel PVQ design algorithm.*

decreased at every iteration in the beginning, but tend to oscillate in a small range afterwards. A typical example is shown in Fig. 3.3. Various strategies may be used to stop the iterations and we adopted the following. The algorithm is run for a specified maximum number of iterations and the best design (resulted in lowest distortion $\bar{D}_{min}$) is saved. However, if the distortion does not drop below $\bar{D}_{min}$ for a specified number of iterations, the algorithm is terminated and the best design is retained. Note that this approach requires additional memory space for storing the best design



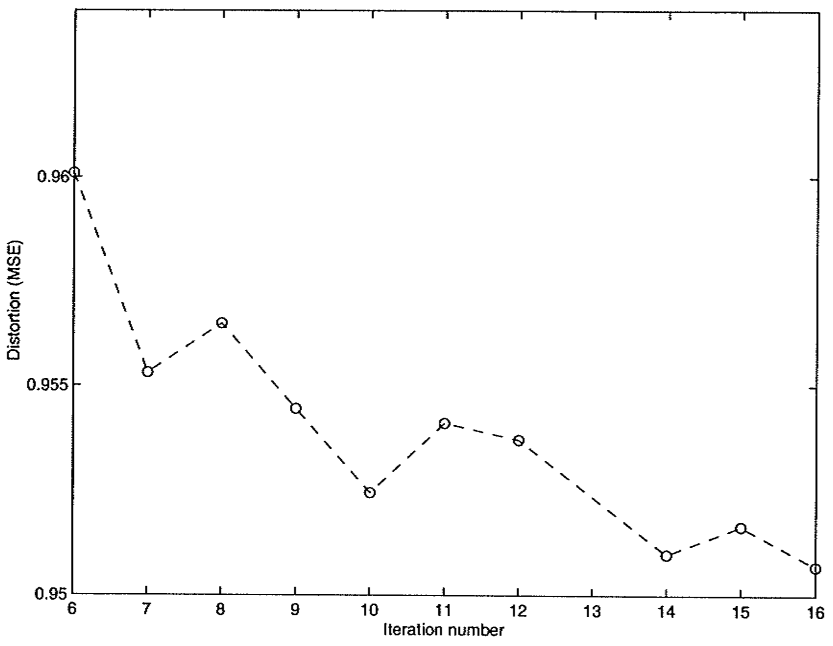


Figure 3.3: *A typical distortion-vs-iteration curve for noisy channel PVQ design algorithm. Left: initially the distortion decreases rapidly, Right: the last 10 iteration at a different scale.*

during a run. In our simulations, the algorithm always resulted in a design which was better than the initial system, and no instabilities of the algorithm were observed.

Another issue is the computational complexity of the algorithm. Out of all the steps in Table 3.1, it is the predictor update step that requires most of the effort. Here, one has to estimate the covariance matrices $\bar{P}$ and $\bar{R}$ and solve a linear system with Pd^2 equations. However, as we have pointed out, this system can usually be solved using efficient solution methods such as *Levinson-Durbin algorithm* [60], [61].

3.8 Soft Decoding

As soft decoding has shown significant improvements in performance when applied to memoryless VQ, it is worth investigating the performance of predictive quantizers with soft-decoding, particularly in the hope that the effect due to error propagation may reduce. The above described algorithm can be extended to handle soft decoding in a straightforward manner. In a noisy channel PVQ system with soft-decoding, the analog channel output $\mathbf{Y}_n$ is directly mapped to the prediction error vectors $\hat{\mathbf{U}}'_n$. Clearly most of the results derived for hard decoding hold true for this case as well, except that we have to replace the discrete channel output $J_n \in \mathbb{I}_N$ with the continuous vector $\mathbf{Y}_n \in \mathbb{R}^L$, where L is the channel dimension. In particular the optimal decoder in (3.12) for prediction error at the receiver now becomes

$$\delta^*_{2_{Soft}} = E\{\mathbf{U}_n | \mathbf{y}_n\}. \tag{3.39}$$

Consequently, the update equations (3.19) and (3.26) have to be modified accordingly. This is straightforward as both ϵ_1 and δ_1 are memoryless (see Ch. 2). Also, the predictor update equations remain valid if the covariance matrices $\bar{P}$ and $\bar{R}$ are estimated based on the outputs of the soft-decoder $\delta^*_{2_{Soft}}$.

3.9 Experimental Results

In this section, the performance of quantizers obtained by the design algorithm presented in Sec. 3.7 is experimentally investigated. In these experiments, predictive VQs are designed for both discrete and waveform channels using the new algorithm, and their performance is compared with that of quantizers designed using several alternative methods.

Signal source- As the signal source, the Gauss-Markov (G-M) process described in Appendix C has been used. This source is commonly used as a benchmark for comparing different source coding techniques as the theoretical bound on performance for this source (rate-distortion function) can be evaluated in many cases, see Appendix C for details. We consider a G-M source with correlation coefficient $\rho = 0.9$ which is typical for real signals such as image and speech data.

Channel- Memoryless additive white Gaussian noise (AWGN) channel model given by the following equation has been used.

$$y_n = s_n + w_n, \tag{3.40}$$

where $s_n = \pm 1$ (antipodal signaling) is the binary input , $y_n \in \mathbb{R}$ is the channel output, and w_n is an iid Gaussian process with mean zero and variance σ_w^2. A soft-decoder directly uses the channel output y_n, while a hard-decoder requires a detector to produce an estimate j_n for the transmitted encoder index i_n. In this case, a maximum a posteriori probability (MAP) detector is designed for the

AWGN channel [62]. Then, the resulting channel can be modeled as a binary symmetric channel (BSC) with error probability p given by [62]

$$p = \frac{1}{2}\mathrm{erfc}(\sqrt{E_s/N_0}), \tag{3.41}$$

where erfc(.) is the complementary error function, E_s is the signal energy, and $N_0 = 2\sigma_w^2$ is the power spectral density of channel noise. The channel transition probabilities can be computed as $p_{ij} = p^{d_H(i,j)}(1-p)^{b-d_H(i,j)}$, $i, j = 1, \ldots, N$, where $d_H(i,j)$ is the Hamming distance between b-bit binary representations of integers i and j. The quality of the BSC is measured by the bit-error rate (BER). In results presented here, both the transition probabilities and BER of the resulting DMC were obtained experimentally from the AWGN channel. When soft-decoding is used, the quality of the channel may be measured by the *channel signal-to-noise ratio* (CSNR)

$$\mathrm{CSNR} = 10\log_{10}(2E_s/N_0). \tag{3.42}$$

Performance measure- signal to noise ratio (SNR) defined below has been used to measure the performance of the quantizers.

$$\mathrm{SNR} = 10\log_{10}\left[\frac{E\|\mathbf{X}_n\|^2}{E\|\mathbf{X} - \hat{\mathbf{X}}'_n\|^2}\right] \;\mathrm{d}B. \tag{3.43}$$

The algorithm introduced in Sec. 3.7 was used to design predictive vector quantizers for various channel noise levels, the performance of which is presented in Figs. 3.4 - 3.7. These designs were carried out and tested using two separate sets of 100,000 source vectors. We have considered PVQs with a transmission rate of 1 bit/sample and vector dimensions of 2, 3, 4, and 5. In these designs, the predictor was confined to be of first-order as in many previous work on PVQ [46], [59], [22]. This is reasonable as the source is a first-order Gauss-Markov process. Note also that the computational complexity and memory requirements grow linearly with the predictor order. In order to apply the given iterative algorithm, initial values for various parameters are required. These were obtained by designing PVQs at the given rate for a noise-free channel, using the closed-loop algorithm in Fig. 4 of [46]. An initial index assignment (IA) for the encoder output (optimized for the channel noise level) was then obtained using the simulated annealing based algorithm described in [40]. This issue will be further discussed later. Finally, the design algorithm Sec. 3.7 was applied to improve the initial PVQ system at the given channel noise level.

A. Main Results

The curves in Figs. 3.4 - 3.7 clearly indicate the advantage of properly designed PVQs over memoryless VQ at the same rate, even in the presence of channel noise. PVQs with hard-decoding achieves about 1-2 dB gain over memoryless COVQ designed for the same channel. The gain is highest at vector dimension of 2 and tends to decrease as the vector dimension is increased. This is due to the fact that, with a blocked scalar process, the correlation between successive vectors diminishes as the block size is increased. Also apparent from these results is the improvement in performance of PVQ with soft-decoding. As expected, this gain increases with the channel noise level. Typically, the systems with soft-decoding achieve about 1 dB gain in SNR compared to the systems with hard-decoding, at high channel noise levels.

Figs. 3.4-3.7 also include the performance of channel optimized PVQ reported in [23] for comparisons. These results have been obtained with a gradient search design

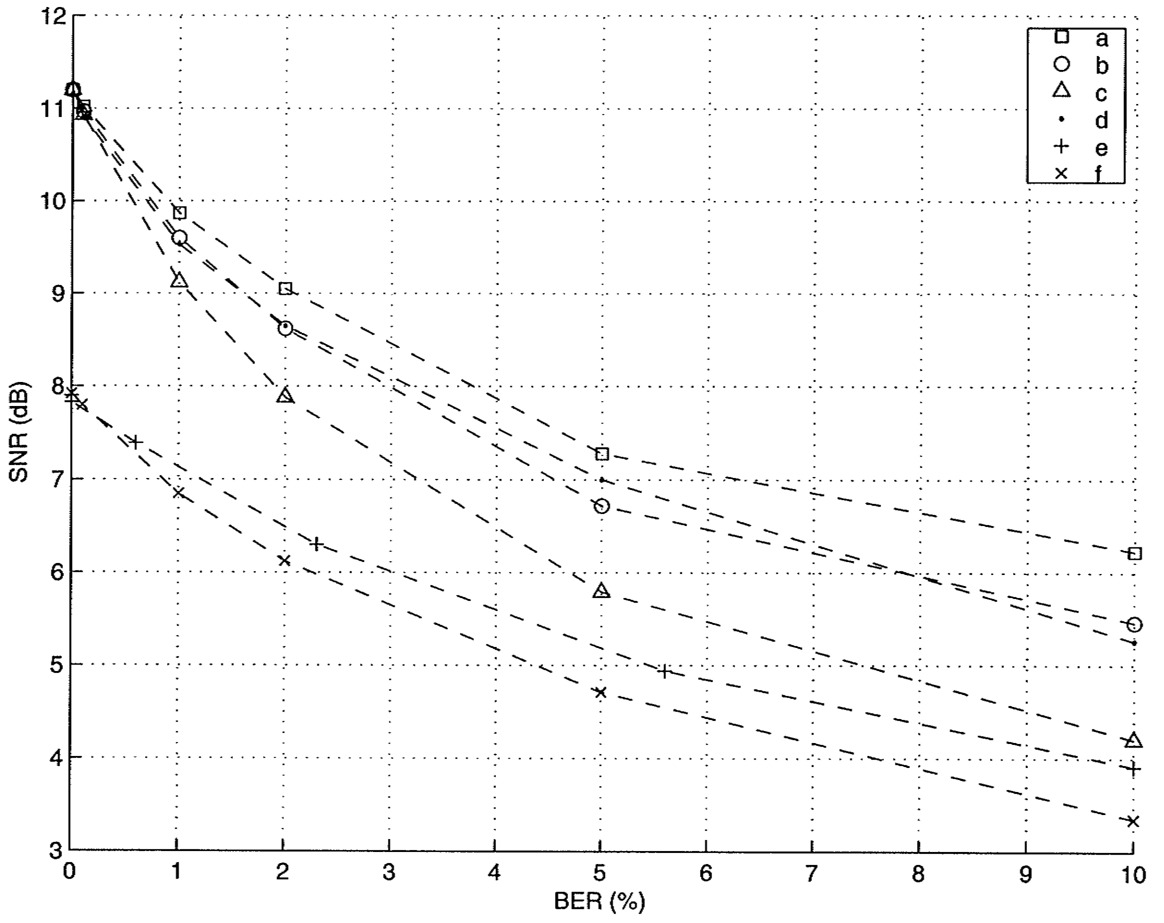


Figure 3.4: *Performance comparison of various noisy channel PVQ schemes for Gauss-Markov source ($\rho = 0.9$) at 1 bit/sample and $d = 2$: (a) proposed algorithm (soft-decoding), (b) proposed algorithm (hard-decoding), (c) proposed algorithm (hard-decoding) without predictor update, (d) COPVQ of [23], (e) memoryless COVQ (soft-decoding), and (f) memoryless COVQ (hard-decoding).*

algorithm, similar to the algorithm of Chang and Gray [59] for noiseless channel PVQ. We note that the performance achievable with our algorithm (with hard-decoding) is nearly identical to those reported in [23] (the gradient search algorithms of [23] are based on hard-decoding and it is not apparent if they can be extended to designing soft-decoding PVQ). A similar observation was also made in [59] regarding the design of PVQs for noise-free channels. That is, the codes obtained with the Lloyd-style algorithm of Cuperman and Gersho [46] yielded a performance almost identical to

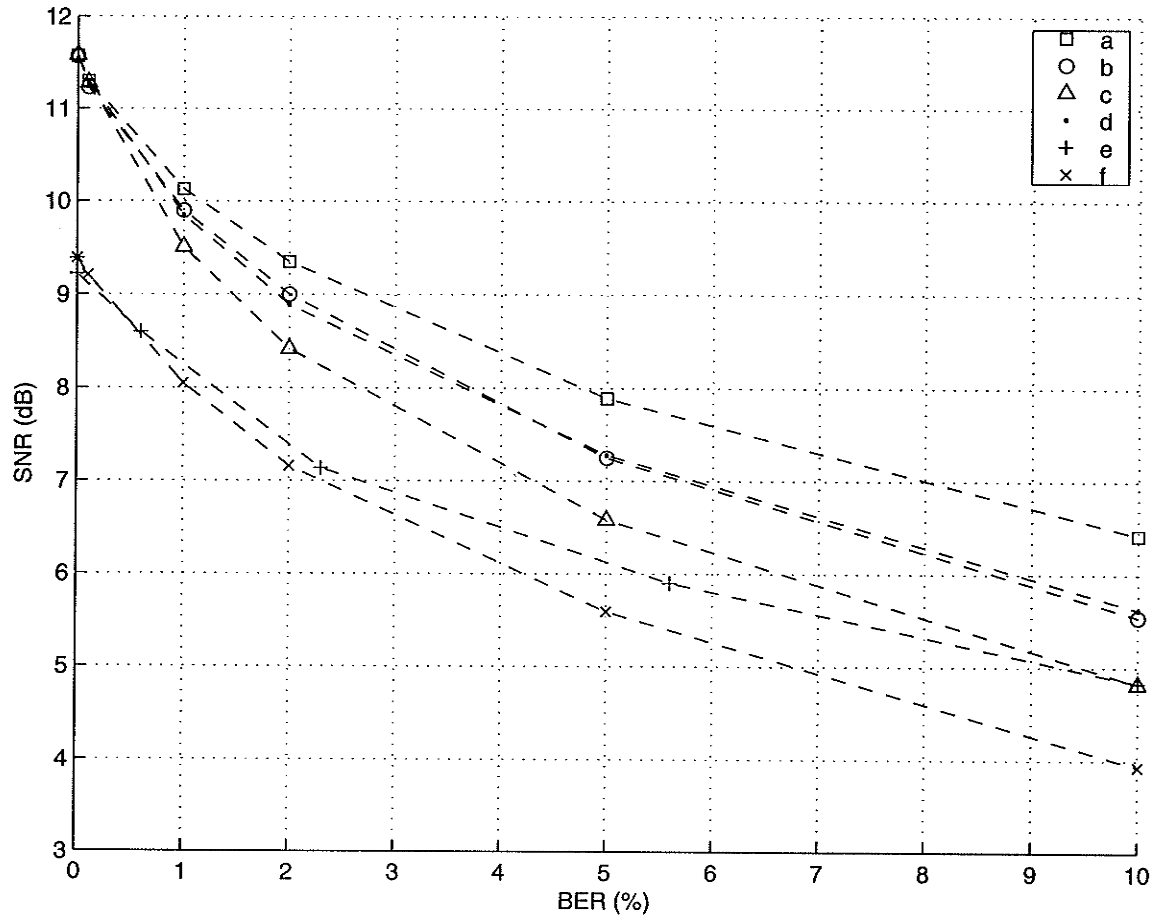


Figure 3.5: *Performance comparison of various noisy channel PVQ schemes for Gauss-Markov source* ($\rho = 0.9$) *at 1 bit/sample and* $d = 3$*: (a) proposed algorithm (soft-decoding), (b) proposed algorithm (hard-decoding), (c) proposed algorithm (hard-decoding) without predictor update, (d) COPVQ of [23], (e) memoryless COVQ (soft-decoding), and (f) memoryless COVQ (hard-decoding).*

those obtained with the gradient search algorithm of Chang and Gray [59]. The latter algorithm also updates the predictor, while the former uses a predictor designed for the unquantized signal. Hence, it was also concluded that the overall performance of PVQ is less sensitive to predictor coefficients when the quantizer is matched to the predictor. We observe below that this does not hold true in the case of noisy channels.

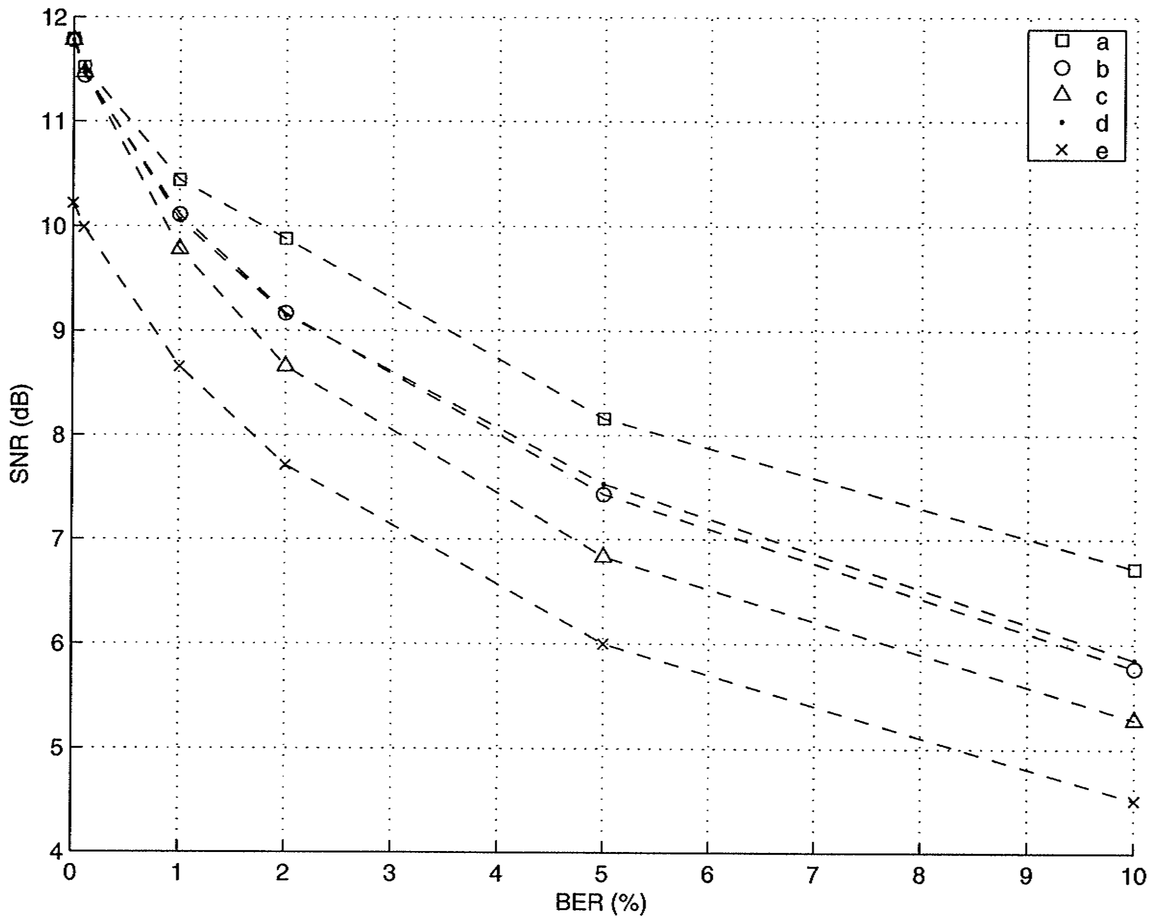


Figure 3.6: *Performance comparison of various noisy channel PVQ schemes for Gauss-Markov source (ρ = 0.9) at 1 bit/sample and d = 4: (a) proposed algorithm (soft-decoding), (b) proposed algorithm (hard-decoding), (c) proposed algorithm (hard-decoding) without predictor update, (d) COPVQ of [23], and (e) memoryless COVQ.*

B. Effect of Predictor Optimization

In Figs. 3.4- 3.7, the curves labeled (c) were obtained by the algorithm in Table 3.1 by omitting the predictor optimization step (*Step 8*), *i.e.*, predictor is not optimized to the channel. The importance of predictor update in our algorithm is evident from these curves. As the channel error rate is increased, the gain in performance achieved by updating the predictor appears very significant. Table 3.2 shows values of predictor coefficients obtained by the proposed algorithm at various channel error

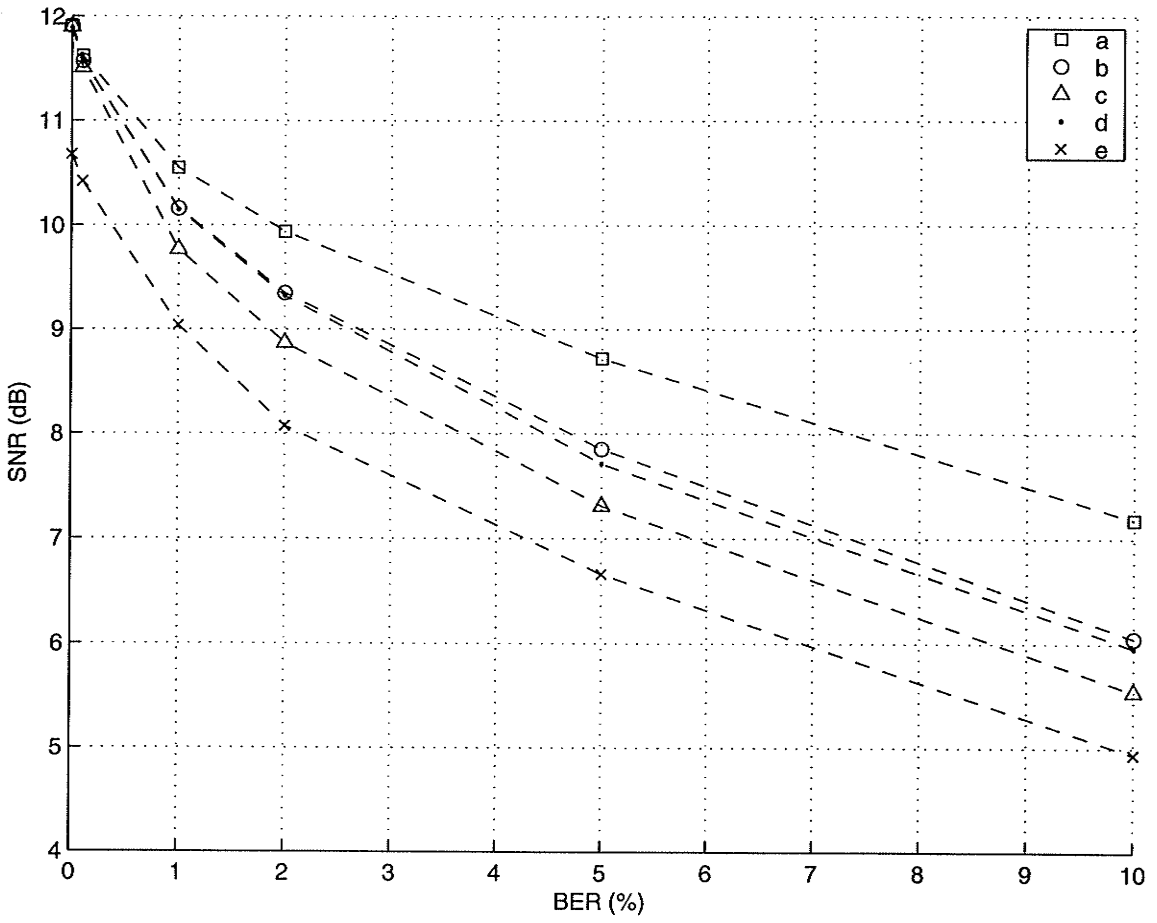


Figure 3.7: *Performance comparison of various noisy channel PVQ schemes for Gauss-Markov source ($\rho = 0.9$) at 1 bit/sample and $d = 5$: (a) proposed algorithm (soft-decoding), (b) proposed algorithm (hard-decoding), (c) proposed algorithm (hard-decoding) without predictor update, (d) COPVQ of [23], and (e) memoryless COVQ.*

rates, in the case of 2-dimensional PVQ. It can be seen that, in the absence of channel noise, most of the prediction is provided by the second component of the predictor input vector ($a_{12} = 0.92$ and $a_{22} = 0.825$), which in our case is the signal sample closest to the two signal samples in the vector being predicted. However, as the channel error rate is increased, the two coefficients a_{12} and a_{22} in the optimized predictor decrease while the remaining two coefficients a_{11} and a_{21} increase. At 10% channel error rate, both components in the predictor input vector seem to contribute

BER (%)	a_{11}	a_{12}	a_{21}	a_{22}
0	0.021	0.920	-0.0144	0.825
0.1	-0.038	0.927	-0.0367	0.834
1	0.005	0.806	0.0353	0.656
2	0.103	0.677	0.148	0.503
5	0.222	0.530	0.301	0.318
10	0.218	0.545	0.287	0.354

Table 3.2: *Variation of predictor coefficients with channel noise level for 2-dimensional PVQ. The predictor coefficients and signal samples are related as follows:* $\tilde{x}_n = a_{11}\hat{x}_{n-3} + a_{12}\hat{x}_{n-2}$ *and* $\tilde{x}_{n-1} = a_{21}\hat{x}_{n-3} + a_{22}\hat{x}_{n-2}$*, where* $\tilde{x}_n$ *is is the predictor output and* $\hat{x}_n$ *is the quantized signal samples at time* n*. In matrix form* $(\tilde{x}_{n-1}\ \tilde{x}_n)^T = A(\hat{x}_{n-3}\ \hat{x}_{n-2})^T$.

similarly to the predicted value. A similar observation was also reported in [23]. An analytical study of the relationship between predictor coefficients and overall mean square error seems difficult in the case of PVQ. However, such a study for differential pulse code modulation (DPCM) was presented in [47] (DPCM is the scalar equivalent of PVQ). There, it was shown that, in the presence of channel errors, the overall MSE of DPCM can be reduced by reducing the prediction gain, that is, by scaling down the predictor coefficient of the first-order linear predictor. Our results show that equivalent conditions hold true in the case of more general PVQ.

C. Effect of Initial IA

We next consider the effect of choosing a good index assignment (IA) for the encoder to initialize the algorithm. In general, the importance of initial conditions used in the Lloyd algorithm depends on the nature of the error surface. In order to investigate the effect of the initial IA on the designs obtained by the proposed algorithm, we compared the performance of PVQs designed by initializing the algorithm with randomly chosen IA against those designed by initializing the algorithm with IA optimized to the channel (using IA algorithm of [40]), and the results are shown in Fig. 3.8 (the

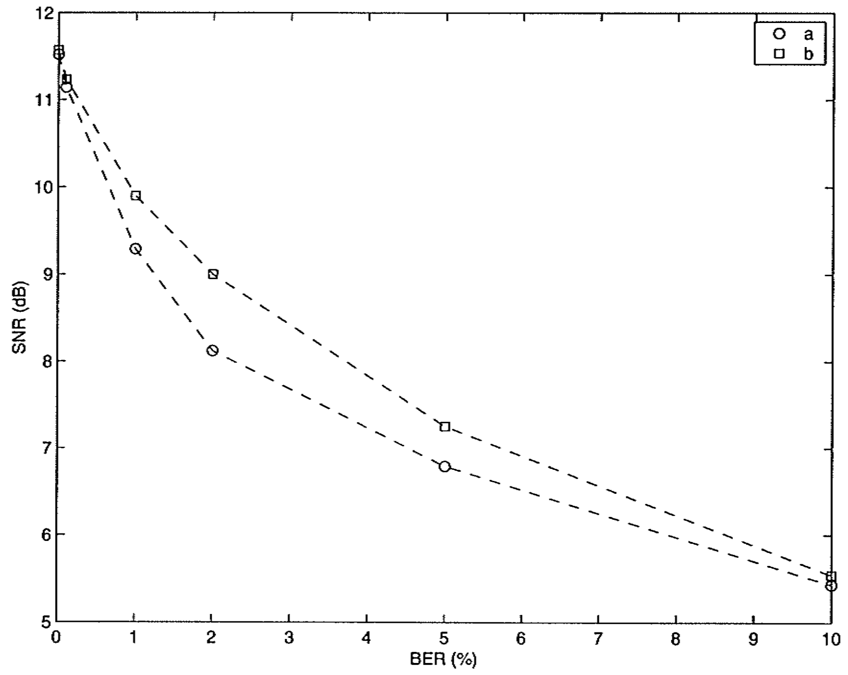


Figure 3.8: *Effect of initial index assignment (N=8, d=3); (a) random IA and (b) optimized IA.*

results have been obtained with hard-decoding). It can be seen that, at some channel error rates the improvement due to good initial IA is significant (about 1 dB at 2 % error rate).

D. Performance Under Channel Mismatch

An explicit assumption in COVQ is that the channel is stationary. However, in real situations this is hardly the case. Furthermore, even if the channel is stationary, one has to estimate the channel parameters through measurements which contain errors. Hence, it is of interest to investigate the robustness of the designs obtained by the proposed algorithm against channel variations. PVQs with rate 1 bits/sample and $d = 4$ were designed using our algorithm, for discrete channels with error probabilities of 1% and 2%. Fig. 3.9 shows the performance of these two PVQs when the channel error

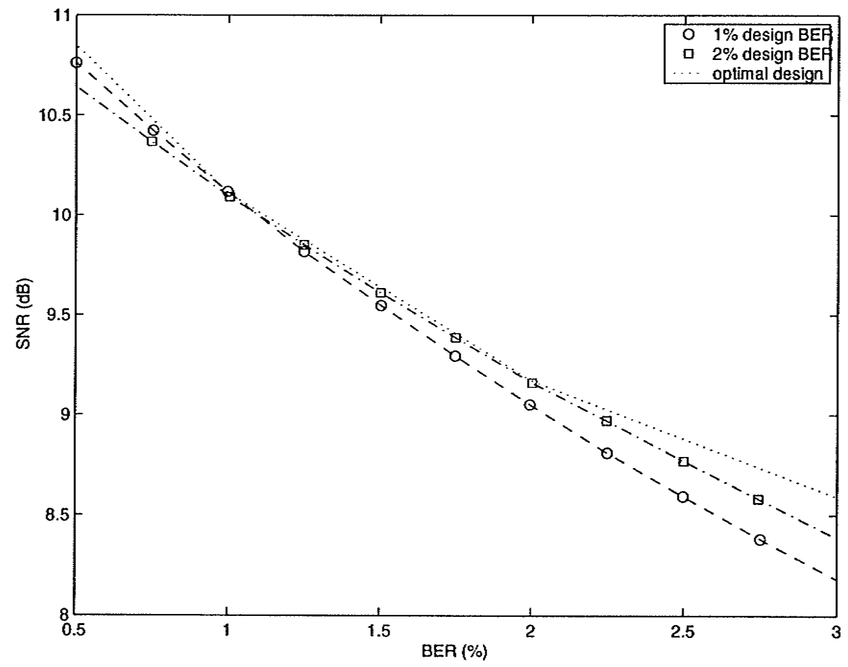


Figure 3.9: *Performance of PVQs designed using the proposed algorithm under channel mismatch. The curve for optimal design correspond to the case when design BER is equal to the actual BER*

is varied in the range of 0.5% - 3%. These results indicate that PVQs designed using the proposed algorithm are robust against moderate variations in the channel error rate. Simulation results in several other experiments also confirmed this observation. In the example shown, both PVQs perform close to the optimal design at channel error rates less than 2%. At 3% channel error rate, the SNR of the PVQ designed for 1% error rate is within 0.5 dB of that achieved with the optimal design.

3.10 Summary

In this chapter, a Lloyd-style iterative algorithm for designing linear prediction-based PVQs for noisy channels was developed. Based on reasonable assumptions, a set of conditions for the optimality of the predictive encoder and the decoder, including the linear predictor were derived. The algorithm can be used to design both hard-decoding and soft-decoding PVQ systems. Experimental results based on Gauss-Markov source and AWGN channel were presented. These results indicate that the PVQs with hard-decoding, produced by the proposed algorithm, perform nearly identical to those obtained in [22], using gradient-search optimization algorithms. Also, it was found that PVQ systems with soft-decoding achieved a gain of about 1.0 dB in overall SNR over hard-decoding systems, when the channel is very noisy.

Chapter 4

Design of Finite-State VQ

4.1 Introduction

A correlated signal source can be quantized more efficiently if the encoder and decoder are chosen based on the signal values observed in the past. A finite-state vector quantizer (FSVQ) is a finite-state machine in which a separate encoder-decoder pair is used in each state for quantizing an input vector, with the state transitions being determined by the observed past of the input signal. Like PVQ discussed in Chapter 3, this approach also gives good performance with relatively small vector dimensions, compared to memoryless VQ operating at the same rate. However, FSVQ is extremely sensitive to channel errors and its performance degrades dramatically under noisy channel conditions. In this chapter, we investigate the problem of designing FSVQ for noisy channels. The main contribution is a robust decoding algorithm for FSVQ operating over noisy channels. An iterative design algorithm for optimizing an FSVQ system to a given channel is also developed.

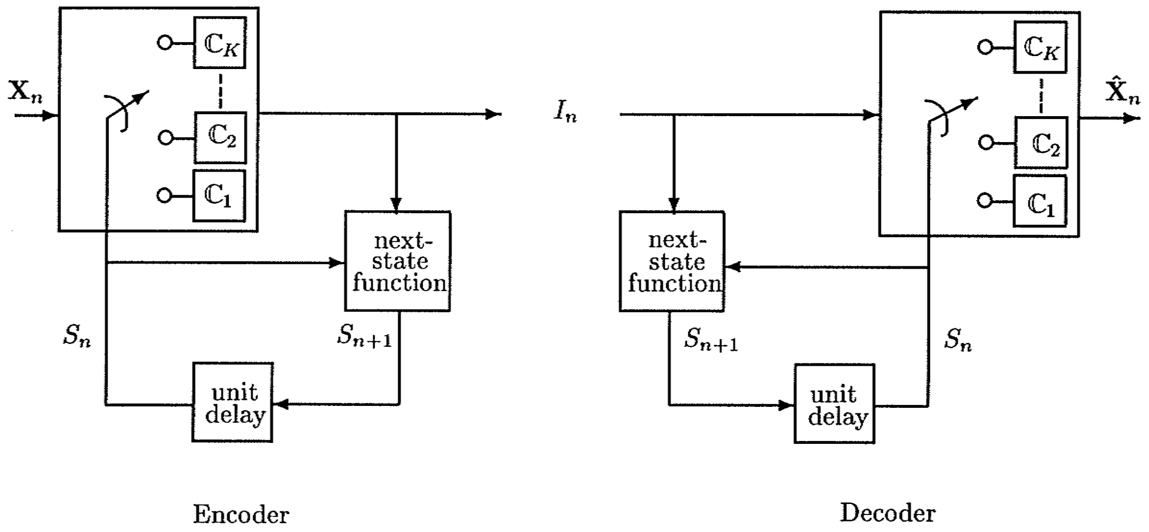


Figure 4.1: *FSVQ encoder and decoder.*

4.2 Finite-State VQ

The idea of FSVQ was first introduced in [49]. In this section, we provide a brief overview of FSVQ. An extensive treatment of the topic can be found in [3]. A conceptual block diagram of an FSVQ system is shown in Fig. 4.1. Let $\mathbf{X}_n \in \mathbb{R}^d$ be a stationary stochastic process. In response to the input sequence, the encoder produces both a sequence of outputs (channel symbols) $I_n \in \mathbb{I}_N = \{1, 2, \ldots, N\}$ and a sequence of *states* $S_n \in \mathbb{S}_K = \{1, 2, \ldots, K\}$, where $n = 0, 1, 2, \ldots$, and $\mathbb{S}_K$ is the *state space.* As can be seen from Fig. 4.1, state S_n is a process with memory and summarizes the dependence of the past inputs on the selection of the current output. In particular, S_n determines which codebook out of K codebooks $\{\mathbb{C}_1, \mathbb{C}_2, \ldots, \mathbb{C}_K\}$ is used to encode $\mathbf{X}_n$. In effect, an FSVQ is a set of memoryless quantizers with a selection rule or a *next-state function* having memory. It is clear that, for an FSVQ to be effective for a certain signal, it's next-state function must predict the best (in the sense of minimizing average distortion) sub-set of N code vectors (state codebook) from a larger collection of code vectors (all K state codebooks), based on the past behavior of the signal. Note the similarity to a predictive quantizer, which can be considered as an FSVQ with an infinite state-space. Also, an FSVQ may be viewed as a PVQ with a non-linear predictor.

An FSVQ is a special case of a more general *finite state code* considered by Gaader and Slepian [63]. A d-dimensional K-state code is specified by three mappings: encoder $\mathcal{E}$, decoder $\mathcal{G}$, and next-state function f as follows;

$$\mathcal{E} : \mathbb{R}^d \times \mathbb{S}_K \to \mathbb{I}_N, \tag{4.1}$$

$$\mathcal{G} : \mathbb{I}_N \times \mathbb{S}_K \to \mathbb{C}, \tag{4.2}$$

$$f : \mathbb{I}_N \times \mathbb{S}_K \to \mathbb{S}_K, \tag{4.3}$$

where $\mathbb{C} = \cup_{i=1}^{K}\mathbb{C}_i$ is the collection of all state codebooks, called the *super codebook.* Given an initial state s_0 and an input sequence $\{\mathbf{x}_n\}$, the encoder produces a channel input sequence $\{i_n\}$ and a state sequence $\{s_n\}$ according to

$$i_n = \mathcal{E}(\mathbf{x}_n, s_n), \tag{4.4}$$

$$s_{n+1} = f(i_n, s_n), \tag{4.5}$$

while the decoder produces the output vector sequence $\{\hat{\mathbf{x}}_n\}$ according to

$$\hat{\mathbf{x}}_n = \mathcal{G}(i_n, s_n), \tag{4.6}$$

where $n = 1, 2, \ldots$. Clearly, given the same initial state, the decoder can track the encoder state sequence, provided that the channel is noiseless (Gaarder and Slepian [63] refer to such a system as "tracking finite-state system"). An FSVQ is a finite-state code with a minimum distortion encoding rule, that is

$$\mathcal{E}(\mathbf{x}_n, s_n) = \arg\min_{i_n} D(\mathbf{x}_n, \mathcal{G}(i_n, s_n)), \tag{4.7}$$

where $D(\mathbf{x}, \hat{\mathbf{x}}) = ||\mathbf{x} - \hat{\mathbf{x}}||^2$ is the square error. The optimal FSVQ is given by [49], [63]

$$\{\mathcal{E}^*, \mathcal{G}^*, f^*\} = \arg \inf_{\mathcal{E},\mathcal{G},f} \lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} ED(\mathbf{X}_i, \hat{\mathbf{X}}_i) \tag{4.8}$$

if the limit exists. The conditions under which this limit exists are mentioned in [49]. In particular, if the input process $\{\mathbf{X}_n\}$ is asymptotically mean stationary (ams), then the joint input-output process of an FSVQ driven by such a source is also ams and the given limit exists.

In general, any FSVQ has two equivalent representations: *labeled-states* (LS-FSVQ) and *labeled transition* (LT-FSVQ) [49] [3]. In the former, every state in the state space has a fixed and distinct code vector associated with it (its label), which becomes the output vector when a transition occurs to that state. In the latter, the code vectors are associated with state transitions so that the same state can produce different outputs, when arrived at from different previous states. Even though these two representations are conceptually equivalent, the codes obtained by iterative code improvement procedures can be very different in each case. The experimental results seem to suggest that the designs based on LT-FSVQ representation perform better than those based on LS-FSVQ representation, though the difference may be considered small [49]. In either case, the critical steps in practical FSVQ design is the selection of the best next-state rule and the state-codebooks. In general, there is no known method for finding even a locally optimal solution to these problems. However, in practice, effective FSVQs (which outperform memoryless VQ) can be designed by using heuristic based procedures. Such a design procedure can be divided into three main steps: (1) designing a"classifier"explicitly for source vectors, (2) obtaining a next-state function and a set of (initial) state codebooks, and (3) iteratively improving the state codebooks. The final step is more or less similar to codebook improvement in memoryless VQ. Several approaches have been investigated for selecting a next-state function, which include conditional histogram design, nearest-neighbor design, and omniscient design [49], [3]. In practice, the omniscient design method has shown to yield best codes in many applications. A brief description of this method is given in Appendix E.

4.3 Noisy channel FSVQ Problem

The successful operation of an FSVQ requires that, given an initial state, the decoder be able to track the sequence of states produced by the encoder. This can be achieved only if the the encoder output codewords can be conveyed to the decoder without any error. In the presence of channel noise, there is a non-zero probability that the decoder receives incorrect codewords. An error in a received codeword will lead to a decoder state sequence that is different to the encoder state sequence, a phenomenon referred to as the "derailing" of the decoder. Since there is only a finite number of states, the decoder will eventually return to the correct state sequence after derailing, provided that no more errors occur in-between. Note however that, during an incorrect state sequence, the decoder essentially picks the output vectors from randomly chosen codebooks. This suggests that the FSVQ will be highly sensitive to channel errors. A comparison of MSE performance of ordinary VQ, PVQ, and FSVQ under noisy channel conditions is shown in Fig. 4.2. Not surprisingly, the performance degradation in FSVQ is much worse than that of PVQ (which in fact performs better

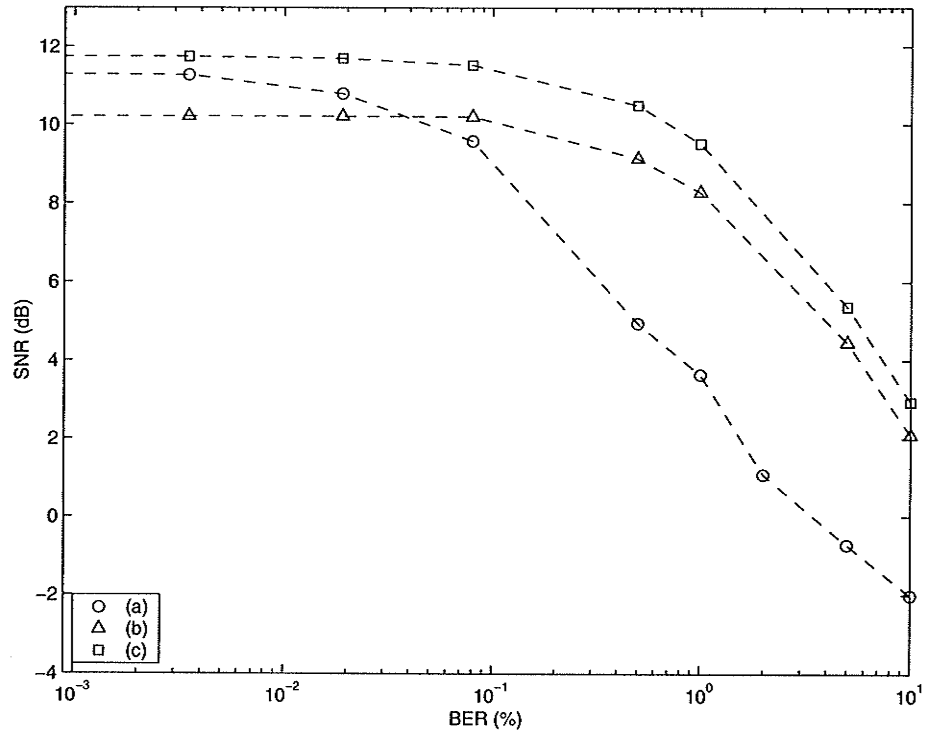


Figure 4.2: *Comparison of performance of (a) FSVQ, (b) ordinary VQ, and (c) PVQ under noisy channel conditions. These results have been obtained with 4-dimensional VQ of Gauss-Markov source (see Appendix C) at* 1 *bit/sample rate, over an AWGN channel. PVQ used* 1st *order linear-prediction while FSVQ had 8 states. In all three systems, index assignment (in the case of FSVQ the algorithm presented in Section 4.8 was used) has been optimized to the channel noise level.*

than memoryless VQ even under noisy channel conditions). In PVQ, a channel error directly affects only the prediction error.

Previously, Hussain and Farvardin [20], [21] studied the design of FSVQ for noisy channels and considered two approaches. In the first (referred to as NC-FSVQ1), the encoder state is explicitly channel coded and transmitted. In order to reduce the additional overhead due to this, the encoder state is transmitted only periodically. The missing states are then estimated using a maximum a posterior sequence detection procedure which requires a delay equal to the intervals at which the encoder state is transmitted. According to the simulation results (for a Gauss-Markov source) reported in [20], this method performed comparably with COVQ when the bit-error rate was less than 5% and outperformed COVQ by 0.4-0.9 dB at the bit error rate of 10%. It appears that this method is effective only when the channel is highly noisy. Furthermore it requires a decoding delay, which can be objectionable in applications such as speech coding, where FSVQ is a strong candidate. In the second approach (referred to as NC-FSVQ2), an FSVQ with a restricted next-state function is designed such that the next-state is solely determined by the previous output of the encoder. In other words, next-state s_{n+1} is contained in the first $\log_2 K$ bits of the channel codeword i_n. In order to obtain a rate of R bits per sample, an additional rate $R - \log_2 K$ bit per sample memoryless VQ is associated with each codeword of the FSVQ. With the chosen next-state function, an error in a received channel codeword affects only the following state, and upon receiving a correct channel codeword the decoder returns to the correct state. The experimental results reported in [20], [21] indicate that the this approach performs better than the former approach as well as memoryless COVQ, at channel error rates in the range of $0.5\% - 10\%$. The restricted next-state rule however has the disadvantage that the number of states K must be less than or equal to N, the number of code vectors per state, so that, for a given vector dimension, reducing the quantizer rate also requires reducing the number of states.

In this dissertation, we propose a new decoder which is robust against channel noise, for an FSVQ with an arbitrary next-state function. The basis of the approach is to view the FSVQ decoding problem as one of selecting the minimum distortion vector $\hat{\mathbf{X}}_n$ from the collection of all state codebooks (*i.e.*, the super codebook), given the sequence of observed channel outputs. In particular, it is shown that the minimum distortion vector can be recursively computed. An important property of this decoder is that it is not a finite-state machine and hence it does not suffer from the derailment problem. It also has the property that, when the channel is noiseless it's performance is identical to that of the FSVQ decoder matched to the encoder. We also present a complete FSVQ design algorithm for noisy channels, based on the proposed decoder. The new decoder can be used with both LS-FSVQ and LT-FSVQ, and the proposed algorithm can be used to iteratively improve (or "channel optimize") any given FSVQ encoder. In our simulations, we used LT-FSVQ encoders and omniscient design procedure to obtain the next-state rule.

4.4 Problem Formulation

A block diagram of a general FSVQ is shown in Fig. 4.3. First, assume that the channel θ is a discrete memoryless channel (DMC), and let i_n and j_n be channel input and output at time n respectively. Then, the channel is described by the mapping $\theta : \mathbb{I}_N \to \mathbb{I}_N$ with transition probabilities $\Pr\{J_n = j_n | I_n = i_n\} = p_{i_n j_n}$. Let (ϵ_s, δ_s) be the encoder and decoder pair associated with the s^{th} state of the FSVQ, where $s = 1, 2, \ldots, K$. Equations 4.4 and 4.6 can now be re-written as

$$i_n = \epsilon_{s_n}(\mathbf{x}_n), \tag{4.9}$$

$$\hat{\mathbf{x}}_n = \delta_{\hat{s}_n}(j_n), \tag{4.10}$$

where $\hat{s}_n$ is the state of the decoder. In the absence of channel noise, $j_n = i_n$ with probability 1 and hence $\hat{s}_n = s_n$. Assume that the initial state of the encoder is s_0, and consider a finite-state decoder which also starts from the state $\hat{s}_0 = s_0$. Let $\bar{J}_n$ denote the channel output sequence $(J_0, J_1, J_2, \ldots, J_n)$. Then, according to the recursive function in (4.5), the decoder state at time n is a deterministic function of

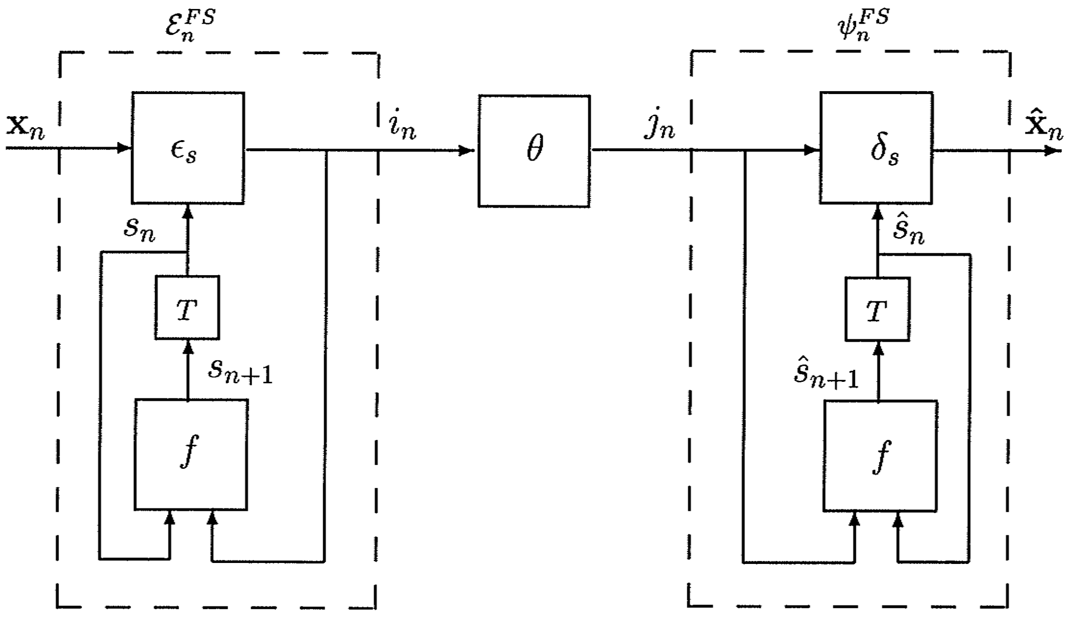


Figure 4.3: *Block diagram of a finite-state vector quantizer (T is a unit delay).*

$\hat{s}_0$ and the channel outputs $\bar{j}_{n-1} = (j_0, j_1, j_2, \ldots, j_{n-1})$, which we explicitly write as $\hat{s}_n(\bar{j}_{n-1}, \hat{s}_0)$. As $n \to \infty$ the decoder state machine becomes stationary and hence

$$\hat{s}_n(\bar{j}_{n-1}, \hat{s}_0) \to \hat{s}_n(\bar{j}_{n-1}) \quad \text{for large } n. \tag{4.11}$$

It is apparent that a finite-state decoder is a special case of a more general class of infinite-memory decoders of the form

$$\hat{\mathbf{x}}_n(j_n) = \psi_n(j_n, \bar{j}_{n-1}) = \psi_n(\bar{j}_n). \tag{4.12}$$

In case of a finite-state decoder, ψ_n has the recursive structure ψ_n^{FS}shown in Fig. 4.3.

In general, the MSE of an FSVQ at time $n \gg 0$ is given by

$$E\{D_n\} = E\|\mathbf{X}_n - \hat{\mathbf{X}}_n\|^2 \;\approx\; E\|\mathbf{X}_n - \psi_n(\bar{J}_n)\|^2$$

$$= \sum_{\bar{j}_n} \int_{\mathbb{R}^d} \|\mathbf{x}_n - \psi_n(\bar{j}_n)\|^2 p(\mathbf{x}_n, \bar{j}_n) d\mathbf{x}_n, \qquad (4.13)$$

where $p(\mathbf{x}_n, \bar{j}_n)$ is the joint density of $\mathbf{X}_n$ and $\bar{J}_n$. As in (4.8), the overall performance of the quantizer is measured by the time averaged MSE

$$\bar{D} = \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} E\{D_i\}, \qquad (4.14)$$

assuming that the limit exists.

The problem at hand is to find the encoder-decoder pair $(\mathcal{E}_n^{FS*}, \psi_n^*)$, which minimizes the performance measure given by (4.14). We will assume that the next-state function f is fixed; a good next-state function for the given source can be found by one of the methods described in [3]. In order to arrive at a Lloyd-style iterative algorithm, we attempt to solve the following two problems:

- Given a decoder ψ_n, determine the optimal finite-state encoder $\mathcal{E}_n^{FS*}$. Since the next-state function is fixed, this is equivalent to finding the optimal encoder ϵ_s^* for each state $s = 1, \ldots, K$.

- Given a finite-state encoder $\mathcal{E}_n^{FS}$, determine the optimal decoder ψ_n^*. If ψ_n is a finite state decoder, this is equivalent to finding an optimal decoder $\delta_{\hat{s}}^*$ for every state $\hat{s} = 1, \ldots, K$. In the case of a general decoder ψ_n, an optimal decoding function has to be found for every n.

4.5 Optimal Decoding for FSVQ

We first consider the optimal finite-state decoder for a given FSVQ encoder. In this case, the MSE as given by (4.13) becomes

$$E\{D_n\} = E\|\mathbf{X}_n - \hat{\mathbf{X}}_n\|^2 = \sum_{\hat{s}_n} \sum_{j_n} \left[\int_{\mathbb{R}^d} \|\mathbf{x}_n - \delta_{\hat{s}_n}(j_n)\|^2 p(\mathbf{x}_n|\hat{s}_n, j_n) d\mathbf{x}_n \right] P(\hat{s}_n, j_n), \tag{4.15}$$

where $\hat{s}_n = f(j_{n-1}, \hat{s}_{n-1})$. For given FSVQ encoder, $E\{D_n\}$ is minimized if the term in the square-bracket is minimized for every j_n and $\hat{s}_n$. The optimal decoder for state $\hat{s}_n = s$ is thus given by

$$\begin{aligned} \delta_s^*(j_n) &= \arg\min_{\delta_s} \int_{\mathbb{R}^d} \|\mathbf{x}_n - \delta_{s_n}(j_n)\|^2 p(\mathbf{x}_n|\hat{s}_n = s, j_n) d\mathbf{x}_n \\ &= E\{\mathbf{X}_n|\hat{s}_n = s, j_n\}, \end{aligned} \tag{4.16}$$

where the last step directly follows from (A.4). Note that when the channel is noise free, $\hat{s}_n = s_n$ and $j_n = i_n$ and the optimal decoder in (4.16) simply gives the centroid of the i_n^{th} cell in the encoder partition of state s_n. The major shortcoming of this decoding scheme in the presence of channel noise is that $j_n \neq i_n$ (due to channel errors) leads to $\hat{s}_{n+1} = f(j_n, \hat{s}_n) \neq s_{n+1}$, which causes the decoder state machine to lose synchronism with the encoder state machine. We next propose a more robust decoder for a given finite-state encoder.

Given a finite-state encoder $\mathcal{E}_n^{FS}$ and a decoder ψ_n, the MSE at time n is given by

$$E\{D_n\} = \sum_{\bar{j}_n} \left[\int_{\mathbb{R}^d} \|\mathbf{x}_n - \psi_n(\bar{j}_n)\|^2 p(\mathbf{x}_n|\bar{j}_n) d\mathbf{x}_n \right] P(\bar{j}_n), \tag{4.17}$$

and it follows that the optimal decoder in this case is given by

$$\begin{aligned}
\psi_n^*(\bar{j}_n) &= \arg\min_{\psi_n} \int_{\mathbb{R}^d} \|\mathbf{x}_n - \psi_n(\bar{j}_n)\|^2 p(\mathbf{x}_n|\bar{j}_n) d\mathbf{x}_n \\
&= E\{\mathbf{X}_n|\bar{j}_n\} \\
&= \sum_{s_n}\sum_{i_n} E\{\mathbf{X}_n|s_n, i_n\} P(s_n, i_n|\bar{j}_n), \qquad (4.18)
\end{aligned}$$

In this expression, $E\{\mathbf{X}_n|s_n, i_n\} = \mathbf{g}_{s_n}(i_n)$ is the centroid of i_n^{th} cell of the encoder partition of the state s_n, which is independent of the channel, where $i_n = 1, 2, \ldots, N$ and $s_n = 1, 2, \ldots, K$. Posterior probabilities $P(s_n, i_n|\bar{j}_n)$ on the other hand are functions of both encoder and channel. Note that if the channel is noiseless, $P(s_n, i_n|\bar{j}_n) = 0$ for all but the correct (s_n, i_n) pair and the decoder output is the centroid of the i_n^{th} cell of the encoder partition of state s_n. We now show that the posterior probabilities $P(s_n, i_n|\bar{j}_n)$ can be recursively computed.

Define the set of all (s_{n-1}, i_{n-1}) pairs which lead to the state $s_n = s$ as $\mu(s) = \{(s_{n-1}, i_{n-1}) : f(s_{n-1}, i_{n-1}) = s\}$. Then

$$\begin{aligned}
P(s_n, i_n|\bar{j}_n) &= \sum_{s_{n-1}}\sum_{i_{n-1}} P(s_n, i_n, s_{n-1}, i_{n-1}|\bar{j}_n) \\
&= \sum_{\mu(s_n)} P(i_n, s_{n-1}, i_{n-1}|\bar{j}_n) \\
&= \frac{1}{P(\bar{j}_n)} \sum_{\mu(s_n)} P(i_n, s_{n-1}, i_{n-1}, \bar{j}_n). \qquad (4.19)
\end{aligned}$$

Now consider

$$\sum_{\mu(s)} P(i_n, s_{n-1}, i_{n-1}, \bar{j}_n) = \sum_{\mu(s_n)} P(j_n|i_n, s_{n-1}, i_{n-1}, \bar{j}_{n-1}) P(i_n, s_{n-1}, i_{n-1}, \bar{j}_{n-1})$$

$$\begin{aligned}
&= P(j_n|i_n)\sum_{\mu(s_n)} P(i_n|s_{n-1},i_{n-1},\bar{j}_{n-1})P(s_{n-1},i_{n-1},\bar{j}_{n-1}) \\
&= P(j_n|i_n)\sum_{\mu(s_n)} P(i_n|s_{n-1},i_{n-1})P(s_{n-1},i_{n-1}|\bar{j}_{n-1})P(\bar{j}_{n-1}) \\
&= \Gamma_n(s_n,i_n)P(\bar{j}_{n-1}), \qquad (4.20)
\end{aligned}$$

where we have used the fact that, for memoryless channel $P(j_n|i_n,s_{n-1},i_{n-1},\bar{j}_{n-1}) = P(j_n|i_n) = p_{i_n j_n}$ and

$$\Gamma_n(s,i) = P(j_n|i_n = i)\sum_{\mu(s)} P(i_n = i|s_{n-1},i_{n-1})P(s_{n-1},i_{n-1}|\bar{j}_{n-1}). \qquad (4.21)$$

Now, noting that

$$\begin{aligned}
P(\bar{j}_n) &= \sum_{s_n}\sum_{i_n}\sum_{\mu(s_n)} P(i_n,s_{n-1},i_{n-1},\bar{j}_n) \\
&= \sum_{s_n}\sum_{i_n}\Gamma_n(s_n,i_n)P(\bar{j}_{n-1}), \qquad (4.22)
\end{aligned}$$

we obtain the desired recursive equation

$$\Lambda_n(s,i) = P(s_n = s, i_n = i|\bar{j}_n) = \frac{\Gamma_n(s,i)}{\sum_{s'=1}^{K}\sum_{i'=1}^{N}\Gamma_n(s',i')}, \qquad (4.23)$$

where

$$\Gamma_n(s,i) = P(j_n|i_n = i)\sum_{\mu(s)} P(i_n = i|s_{n-1},i_{n-1})\Lambda_{n-1}(s,i). \qquad (4.24)$$

In (4.24), the probabilities $P(i_n|s_{n-1},i_{n-1})$ depend only on source statistics and the encoder, and hence can be computed for a given source and an encoder. The optimal decoder given by (4.18) can now be computed recursively, as shown in Table 4.1. A block diagram of the decoder is given in Fig. 4.4.

A noteworthy feature of the proposed decoder is that it is not a finite-state ma-

Initialization:	Assign initial values to $\Lambda_0(s,i)$, $s = 1, 2, \ldots, K$ and $i = 1, 2, \ldots, N$ and $n \leftarrow 1$.
Step 1:	Given the observed channel output j_n compute $\Gamma_n(s,i)$, $s = 1, 2, \ldots, K$ and $i = 1, 2, \ldots, N$ using (4.24).
Step 2:	Compute $\Lambda_n(s,i)$ $s = 1, 2, \ldots, K$ and $i = 1, 2, \ldots, N$. using (4.23).
Step 3:	Compute decoded output as $\hat{\mathbf{x}}_n = \sum_{s=1}^{K} \sum_{i=1}^{N} \mathbf{g}_s(i) \Lambda_n(s,i)$.
Step 4:	$n \leftarrow n + 1$ and goto Step 1.

Table 4.1: *Recursive computation of FSVQ decoder in (4.16).*

chine. It may be viewed as an infinite-state machine, with $P(s_n, i_n | \bar{j}_n)$, $s_n = 1, \ldots, K$ and $i_n = 1, \ldots, N$ being the state (see Fig. 4.4). In the absence of channel noise, the optimal reproduction vector for i_n^{th} cell in the encoder partition of the s_n^{th} state is given by $E\{X_n | s_n, i_n\}$ (centroid condition). Note that decoder in that case observes i_n and is able to derive the exact value of s_n from the sequence $i_{n-1}, i_{n-2}, \ldots$. When the channel is noisy, the decoder observes only a noisy version j_n of i_n and the exact value of s_n cannot be determined. Hence, the optimal decoding rule, in the sense of minimizing the MMSE, is to compute the most likely value of $E\{X_n | s_n, i_n\}$, given the channel output sequence $j_n, j_{n-1}, \ldots$, as given by (4.18). Instead of recursively computing the encoder state s_n, the decoder recursively computes the posterior probabilities $P(s_n, i_n | j_n, j_{n-1}, \ldots)$. It is also worth noting that this decoding algorithm assumes a general FSVQ encoder and hence is applicable to both labeled-state and labeled-transition types.

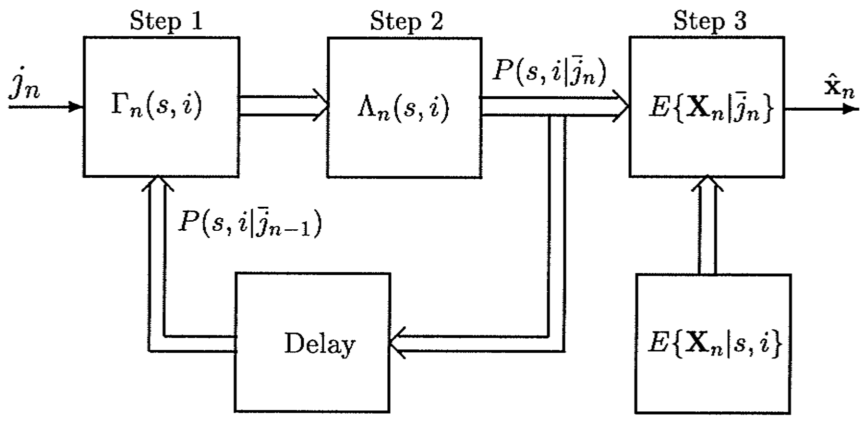


Figure 4.4: *A block-diagram representation of the recursive decoder in (4.16). It may be viewed as an infinite-state machine with the state $P(s_n, i_n|\bar{j}_n)$, $s_n = 1,\ldots,K$ and $i_n = 1,\ldots,N$.*

We next consider the computational complexity and storage requirements of the decoding algorithm shown in Table 4.1. Step 1 requires $N\sum_{l=1}^{K}|\mu(s_l)|$ computations, where $|\mu(s_l)|$ is the cardinality of the set $\mu(s_l)$. Since there are N possible transitions from each of the K states, $\sum_{l=1}^{K}|\mu(s_l)| = KN$. Hence, the algorithm requires (ignoring the normalization in Step 2) $KN^2 + KN \approx KN^2$ computations for $N >> 1$. The algorithm requires the storage of the set of all encoder centroids $\mathbf{g}_s(i)$, $s = 1,2,\ldots,K$, $i = 1,2,\ldots,N$ and the probabilities $P(i_n|s_{n-1}, i_{n-1})$ $i_{n-1}, i_n = 1,2,\ldots,N$ and $s_{n-1} = 1,\ldots,K$. Hence, the total storage requirement in terms of floating-point variables is $KN^2 + KN \approx KN^2$ for $N >> 1$.

4.6 Optimal Encoder

The optimal encoder partition for each state, given a next-state rule and the decoder derived above is considered next. Let $\epsilon_s(\mathbf{x})$ be the encoder associated with the state s. That is

$$\epsilon_s(\mathbf{x}) = i \Leftrightarrow \mathbf{x} \in \Omega_s(i), \tag{4.25}$$

where $\cup_{i=1}^{N}\Omega_s(i) = \mathbb{R}^d$ and $\cap_{i=1}^{N}\Omega_s(i) = \emptyset$. Such a mapping of course has to be chosen so that the average distortion of the resulting system is minimized. To this end, consider the MSE of the system with a finite-state encoder and the decoder derived in the previous section. From (4.13), MSE for $n \gg 0$ can be expressed as

$$\begin{aligned}
E\{\|\mathbf{X}_n - \hat{\mathbf{X}}_n\|^2\} &\approx \int_{\mathbb{R}^d} \sum_{\bar{j}_n} \|\mathbf{x}_n - \psi_n(\bar{j}_n)\|^2 P(\mathbf{x}_n, \bar{j}_n) d\mathbf{x}_n \\
&= \sum_{s_n, i_n} \int_{\mathbb{R}^d} \sum_{\bar{j}_n} \|\mathbf{x}_n - \psi_n(\bar{j}_n)\|^2 P(\mathbf{x}_n, \bar{j}_n | s_n, i_n) P(s_n, i_n) d\mathbf{x}_n \\
&= \sum_{s_n, i_n} \int_{\mathbb{R}^d} \sum_{\bar{j}_n} \|\mathbf{x}_n - \psi_n(\bar{j}_n)\|^2 P(\bar{j}_n | s_n, i_n) p(\mathbf{x}_n | s_n, i_n) P(s_n, i_n) d\mathbf{x}_n \\
&= \sum_{s_n, i_n} P(s_n) \int_{\Omega_{s_n}(i_n)} \sum_{\bar{j}_n} \|\mathbf{x}_n - \psi_n(\bar{j}_n)\|^2 P(\bar{j}_n | s_n, i_n) p(\mathbf{x}_n | s_n) d\mathbf{x}_n \\
&= \sum_{s_n, i_n} P(s_n) \int_{\Omega_{s_n}(i_n)} E_{\hat{\mathbf{X}}_n}\{\|\mathbf{x}_n - \hat{\mathbf{X}}_n\|^2 | s_n, i_n\} p(\mathbf{x}_n | s_n) d\mathbf{x}_n,
\end{aligned} \tag{4.26}$$

where we have used the fact that

$$p(\mathbf{x}_n|s,i) = \begin{cases} p(\mathbf{x}_n|s)/p(i|s) & \mathbf{x}_n \in \Omega_s(i) \\ 0 & \text{elsewhere.} \end{cases} \tag{4.27}$$

Given $\hat{\mathbf{x}}_n = \psi_n(\bar{j}_n)$, the optimal partition for state s at time n is given by

$$\epsilon_s^*(\mathbf{x}_n) = \arg\min_i E\{\|\mathbf{x}_n - \hat{\mathbf{X}}_n\|^2 | s, i\}. \tag{4.28}$$

However, as the decoder ψ_n is time-varying, the optimal encoder for state s obtained in this manner is also time-varying. In order to obtain a time-invariant encoder partition for every state s, we define the distortion measure to be minimized as the time average

$$\bar{D}_{\mathbf{x},s}(i) = \lim_{l\to\infty} \frac{1}{l} \sum_{n=1}^{l} E\{\|\mathbf{x}_n - \hat{\mathbf{X}}_n\|^2 | s_n = s, i_n = i, \mathbf{x}_n = \mathbf{x}\}, \tag{4.29}$$

assuming that the limit exists. This limit exists if s_n is an ergodic Markov chain. Then, the optimal encoder partition for state s is described by

$$\begin{aligned} \epsilon_s^*(\mathbf{x}_n) = i \iff & \lim_{l\to\infty} \frac{1}{l} \sum_{n=1}^{l} E\{\|\mathbf{x}_n - \hat{\mathbf{X}}_n\|^2 | s_n = s, i_n = i\} \\ & \leq \lim_{l\to\infty} \frac{1}{l} \sum_{n=1}^{l} E\{\|\mathbf{x}_n - \hat{\mathbf{X}}_n\|^2 | s_n = s, i_n = k\} \;\; \forall\, k \neq i \end{aligned} \tag{4.30}$$

for every $\mathbf{x}_n \in \mathbb{R}^d$. Define

$$\begin{aligned} \mathbf{a}_i(s) &= \lim_{l\to\infty} \frac{1}{l} \sum_{n=1}^{l} E\{\hat{\mathbf{X}}_n | s_n = s, i_n = i\}, \\ b_i(s) &= \lim_{l\to\infty} \frac{1}{l} \sum_{n=1}^{l} E\{\|\hat{\mathbf{X}}_n\|^2 | s_n = s, i_n = i\}, \quad i = 1, \ldots, N, \;\; s = 1, \ldots, K. \end{aligned} \tag{4.31}$$

The optimal encoder for state s can be given in the form

$$\epsilon_s^*(\mathbf{x}_n) = i \iff b_i(s) - 2\mathbf{a}_i^T(s)\mathbf{x}_n \leq b_k(s) - 2\mathbf{a}_k^T(s)\mathbf{x}_n \;\; \forall\, k \neq i. \tag{4.32}$$

Since the encoder output sequence depends on the past inputs, the parameters $\{\mathbf{a}_i(s), b_i(s)\}$, $i = 1, \ldots, N$ and $s = 1, \ldots, K$ cannot in general be computed without considering an entire sequence. In the iterative design procedure to be presented shortly, these parameters may be computed using training sequences as follows. Let

$\{i_n^{(t-1)}\}$ and $\{s_n^{(t-1)}\}$ be the encoder output and state sequence respectively, generated in the $(t-1)^{th}$ iteration. Then, the values of encoder parameters (given the decoder) in t^{th} iteration is computed as

$$\begin{aligned} \hat{\mathbf{a}}_i^{(t)}(s) &= \frac{1}{n_T L_{is}} \sum_{s,i} \sum_{n=1}^{n_T} \hat{\mathbf{x}}_n^{(t)} \mathbf{1}(s_n^{(t-1)} = s, i_n^{(t-1)} = i), \\ \hat{b}_i^{(t)}(s) &= \frac{1}{n_T L_{is}} \sum_{s,i} \sum_{n=1}^{n_T} \|\hat{\mathbf{x}}_n^{(t)}\|^2 \mathbf{1}(s_n^{(t-1)} = s, i_n^{(t-1)} = i), \end{aligned} \tag{4.33}$$

where $\mathbf{1}(.)$ is the indicator function, L_{is} is the number of times the state s and the index i occurred at the same time in the sequences $\{i_n^{(t-1)}\}$ and $\{s_n^{(t-1)}\}$, and n_T is the size of the training sequence representing the source.

4.7 Design Algorithm

The encoder and decoder derived above can be used to iteratively design a complete channel optimized FSVQ from a training set of source vectors. The basic algorithm follows the same philosophy as the GLA. In this thesis, we have not considered the optimization of the next state-rule to the noisy channel conditions. Instead, a given finite state encoder (with a fixed next state rule) is optimized to the channel. At this point we note that the optimality conditions used here do not guarantee a locally optimal solution as in GLA, *i.e.*, we do not necessarily obtain a monotonically decreasing sequence of distortion. However, the experimental results presented here indicate that, in terms of convergence properties, the algorithm exhibits a behavior similar to that of the noisy channel PVQ design algorithm described in Chapter 3. More importantly, we demonstrate that the finite state codes designed by this algorithm are robust against channel noise and give a performance superior to that can be achieved with memoryless COVQ operating over the same channel, at the same rate.

Let $\{\mathbf{x}_n\}_{n=1}^{n_T}$ be a training set drawn from the source. The quantizer is designed to minimize the empirical MSE given by

$$\hat{D} = \frac{1}{n_M n_T d} \sum_{m=1}^{n_M} \sum_{n=1}^{n_T} \|\mathbf{x}_n - \hat{\mathbf{x}}_n(\bar{\jmath}_{nm}, \Lambda_0)\|^2, \tag{4.34}$$

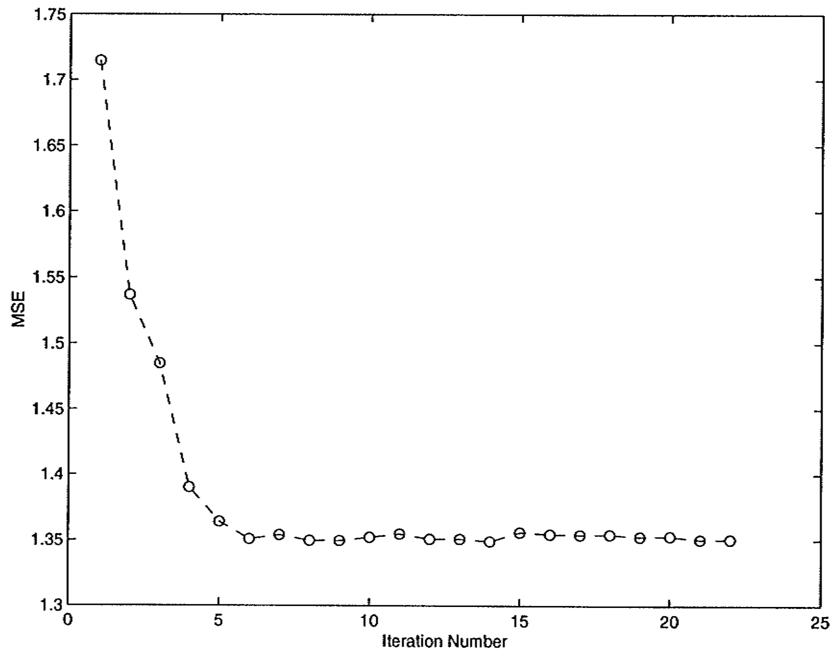
where $\bar{\jmath}_{nm} = (\jmath_{1m}, \jmath_{2m}, \ldots, \jmath_{nm})$ and $\{\jmath_{nm}\}_{n=1}^{n_T}$ is the channel output sequence obtained by transmitting (over a simulated channel) the encoder output sequence for the m^{th} time, $m = 1, \ldots, n_M$ and Λ_0 is the initial state of the decoder. In order to start the algorithm, we require an initial encoder, *i.e.*, a next-state rule, a set of encoder partitions for each state, and an index assignment (IA). While one could use an arbitrary IA, a method of obtaining a better choice is presented in the next section. In each iteration of the algorithm, the encoder partitions and the decoder are changed such that the average distortion is reduced. A next state-rule (which will not be changed by the algorithm) and a set of initial state-encoders can be obtained by any of the approaches described in [49]. However, the performance of the final code resulting from different methods may differ considerably. In our experiments, we used the omniscient labeled transition (OLT-FSVQ) method as it has shown to yield best codes for noiseless channels [49]. For the sake of completeness, a brief synopsis of OLT-FSVQ design procedure as we used it here, is given in Appendix E. Let $\{\mathbf{a}_i^{(0)}(s), b_i^{(0)}(s)\}$, $i = 1, \ldots, N$, $s = 1, \ldots, K$ be the parameters of initial state encoders. Also let the superscript (k) denote the values of the various parameters in the k^{th} iteration. The complete design algorithm is presented in Table 4.2.

A key issue related to the given algorithm is its convergence. We have not attempted to prove the convergence of the algorithm and we note that the arguments used in the context of ordinary GLA cannot be used with iterative design algorithms

```
Step 0: Given: ${\mathbf{a}_i^{(0)}(s),\, b_i^{(0)}(s)}_{i=1}^{N}$, and training set ${\mathbf{x}_n}_{n=1}^{n_T}$.
        Compute encoder output sequence ${i_n^{(0)}}_{n=1}^{n_T}$,
        decoder parameters $P^{(0)}(i_n|s_{n-1}, i_{n-1})$, and $\mathbf{g}_s^{(0)}(i)$,
        $i_n, i_{n-1}, i = 1, \ldots, N,\ s_n, s = 1, \ldots, K$.
        $k \leftarrow 0$.
Step 1: Compute distortion $\hat{D}^{(k)}$ using (4.34).
        If convergence criteria are satisfied stop; Else let $k \leftarrow k + 1$.
Step 2: Compute encoder parameters ${\mathbf{a}_i^{(k)}(s),\, b_i^{(k)}(s)}$,
        $i = 1, \ldots, N,\ s = 1, \ldots, K$ using (4.33).
Step 3: Compute decoder parameters $P^{(k)}(i_n|s_{n-1}, i_{n-1})$ and $\mathbf{g}_s^{(k)}(i)$,
        $i_n, i_{n-1}, i = 1, \ldots, N,\ s_n, s = 1, \ldots, K$.
Step 4: Update encoder output sequence ${i_n^{(k)}}_{n=1}^{n_T}$.
        Repeat from Step 1.
```

Table 4.2: *Proposed noisy channel FSVQ design algorithm.*

for FSVQ [49]. However, to our satisfaction, the algorithm always appeared to converge, at least in mean, to a local minimum in all our simulations. The variation of distortion in a typical run of the algorithm is shown in Fig. 4.5. This behavior is similar to that of the noisy channel PVQ algorithm described in Section 3.7 (see Fig. 3.3). We can use similar criteria to stop our FSVQ algorithm as well, which proved to be quite successful in our simulations.



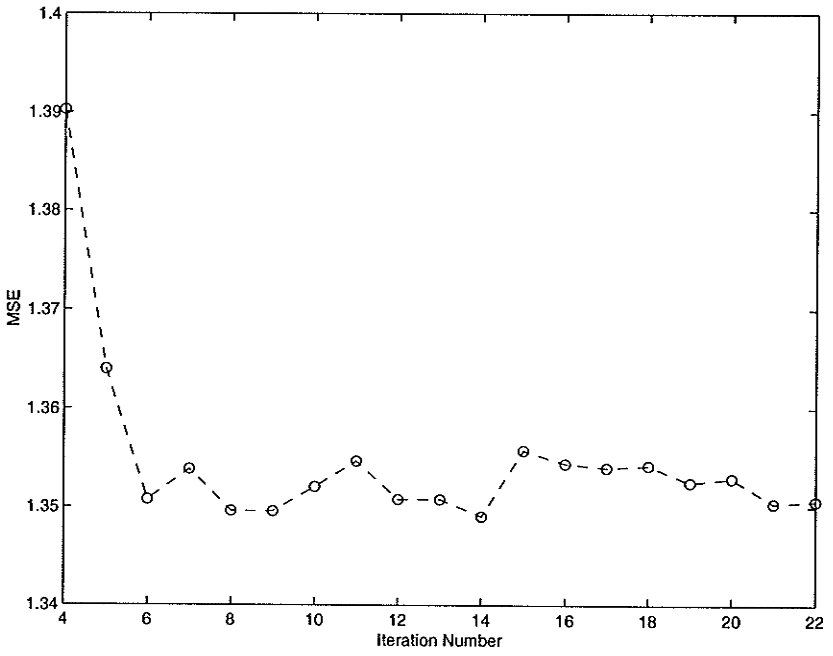


Figure 4.5: *A typical distortion-vs-iteration curve for the noisy channel FSVQ design algorithm given in Table 4.2. Top: initially the distortion decreases rapidly, Bottom: the latter part of the curve at a different scale.*

4.8 Index Assignment

In this section, we consider the problem of good IA in the context of FSVQ. Such an IA can be used to initialize the FSVQ design algorithm presented in Table 4.2. The main result obtained in this section is a distortion measure which can be used with the IA algorithm presented in [40]. The basic idea is to separate channel distortion from quantization error, and to solve the combinatorial optimization problem of determining the mapping from the set of encoder indices to the set of channel codewords, which results in minimum average channel distortion. Deriving an appropriate channel distortion measure for the recursive decoder introduced above is difficult. Hence, we find instead the optimal IA for an ordinary FSVQ decoder, which is a much simpler problem. It can be expected that such an IA will also be effective with the decoder proposed above.

Consider the FSVQ system shown in Fig. 4.3. It can be shown that (see Appendix D), the overall MSE of the the system can be written as

$$E\|\mathbf{X}_n - \hat{\mathbf{X}}_n\|^2 = D_Q + D_C, \tag{4.35}$$

where

$$D_Q = \sum_{s_n, i_n} P(s_n) \int_{\Omega_{s_n}(i_n)} \|\mathbf{x}_n - \mathbf{g}_{s_n}(i_n)\|^2 p(\mathbf{x}_n|s_n) d\mathbf{x}_n, \tag{4.36}$$

$$D_C = \sum_{s_n, i_n} \sum_{\hat{s}_n, j_n} P(\hat{s}_n|s_n) P(j_n|i_n) P(s_n) \|\mathbf{g}_{s_n}(i_n) - \hat{\mathbf{x}}_n(\hat{s}_n, j_n)\|^2 P_{s_n}(i_n), \tag{4.37}$$

and $\mathbf{g}_s(i)$ is the centroid of the i^{th} encoding cell $\Omega_s(i)$ of the s^{th} state, $s = 1, \ldots, K$ and $i = 1, \ldots, N$. D_Q represents the average MSE in approximating $\mathbf{x}_n \in \Omega_{s_n}(i_n)$ by

$\mathbf{g}_{s_n}(i_n)$ and hence can be considered as the quantization error, which is independent of the channel. D_C on the other hand is the MSE due to using $\hat{\mathbf{x}}_n(\hat{s}_n, j_n)$ instead of $\mathbf{g}_{s_n}(i_n)$ for the reconstruction of $\mathbf{x}_n \in \Omega_{s_n}(i_n)$, and is therefore can be considered as the contribution from channel noise to the overall MSE. Clearly, only the latter is affected by the assignment of channel codewords, that is both $P(\hat{s}_n|s_n)$ and $P(j_n|i_n)$ depend on IA.

In order to formulate the IA problem here, we view the FSVQ encoder as a mapping of the state-index pair (s, i), $s = 1, \ldots, K$ and $i = 1, \ldots, N$, to a channel code consisting of N binary codewords, *i.e.*, $\mathbb{S}_K \times \mathbb{I}_N \to \mathbb{B}_N$, where $\mathbb{B}_N = \{b_1, b_2, \ldots, b_N\}$ is the channel code. In other words, we have to find an index assignment $\pi(s, i)$ such that, if $\pi(s, i) = m$, then $b_m \in \mathbb{B}$ is the channel codeword transmitted for index i of state s. Given some $\pi(s, i)$, the conditional probability $P(j_n|i_n)$ in (4.37) can be written as

$$P(j_n|i_n) = p_{\pi(s_n, in)\pi(\hat{s}_n, j_n)}, \tag{4.38}$$

where p_{kl}, $k, l = 1, \ldots, N$, are the channel transition probabilities. We note that there are $N!$ different ways of mapping the index set of each state encoder to the channel code and therefore $K(N!)$ different possibilities for $\pi(s, i)$. The problem here is to choose, out of these $K(N!)$ possibilities, the one which minimizes D_C in (4.37). It is practically impossible in most cases to find the optimal mapping through an exhaustive search. A popular method for solving the IA problem is by using simulated-annealing. The idea was first proposed by Farvardin [40] for memoryless VQ. We adapt the same algorithm for FSVQ with two minor modifications; (i) the objective function is replaced by D_C in (4.37), (ii) the "state" of the system being annealed (completely different to the encoder state!) is defined as the choice of the mapping π and a perturbation of this state is defined as an interchange of two randomly chosen indices in a randomly chosen encoder state s. Other details of the algorithm can be found in [40].

4.9 Soft-decoding

The recursive decoder and the FSVQ design algorithm presented in the previous sections can be extended to soft-decoding with simple modifications. It is of interest to investigate the performance improvements that could be obtained by using analog channel outputs in the proposed recursive decoder. In particular, the soft-decoding can further reduce the effect of error propagation in the receiver.

Let $\mathbf{Y}_n \in \mathbb{R}^L$ denote the channel output vector at time n (see Section 2.2) and let $\bar{\mathbf{Y}}_{n-1} = (\mathbf{Y}_1, \ldots, \mathbf{Y}_{n-1})$. Then, based on the result in (4.18), we can write the optimal soft decoder as

$$\begin{aligned} E\{\mathbf{X}_n|\hat{s}_n, \mathbf{y}_n\} &= E\{\mathbf{X}_n|\bar{\mathbf{y}}_n\} \\ &= \sum_{s_n}\sum_{i_n} E\{\mathbf{X}_n|s_n, i_n\}P(s_n, i_n|\bar{\mathbf{y}}_n). \end{aligned} \tag{4.39}$$

The derivation of the iterative decoding algorithm follows the same steps as in the hard decoding case, with the exception of the channel transition probabilities being replaced by the conditional densities $p_i(\mathbf{y})$, defined in (2.12), $i = 1, \ldots, N$. Hence, the algorithm in Table 4.1 is still applicable with the re-definition of Γ_n and Λ_n as follows:

$$\Lambda_n(s,i) = P(s_n = s, i_n = i|\bar{\mathbf{y}}_n), \tag{4.40}$$

$$\Gamma_n(s,i) = p_i(\mathbf{y}_n)\sum_{\mu(s)} P(i_n = i|s_{n-1}, i_{n-1})\Lambda_{n-1}(s,i). \tag{4.41}$$

The MSE of the system at time n is given by

$$E\{\|\mathbf{X}_n - \hat{\mathbf{X}}_n\|^2\} = \sum_{s_n, i_n} P(s_n) \int_{\Omega_{s_n(i_n)}} E_{\hat{\mathbf{X}}_n}\{\|\mathbf{x}_n - \hat{\mathbf{x}}_n\|^2 | s_n, i_n\} p(\mathbf{x}_n | s_n) d\mathbf{x}_n, \quad (4.42)$$

where $\hat{\mathbf{x}}_n = \sum_s \sum_i \mathbf{g}_s(i) \Lambda_n(s,i)$ and

$$E_{\hat{\mathbf{X}}_n}\{\|\mathbf{x}_n - \hat{\mathbf{x}}_n\|^2 | s_n, i_n\} = \int_{\mathbb{R}^d} \|\mathbf{x}_n - \hat{\mathbf{x}}_n\|^2 p(\hat{\mathbf{x}}_n | s_n, i_n) d\hat{\mathbf{x}}_n. \quad (4.43)$$

The decoder parameters are still given by (4.33) and the empirical distortion is given by

$$\hat{D} = \frac{1}{n_M n_T d} \sum_{m=1}^{n_M} \sum_{n=1}^{n_T} \|\mathbf{x}_n - \hat{\mathbf{x}}_n(\bar{\mathbf{y}}_{nm}, \Lambda_0)\|^2. \quad (4.44)$$

4.10 Experimental Results

In this section, we investigate experimentally the performance of FSVQ designed using the algorithm introduced in Section 4.7. The source, channel, and performance measure used in these experiments are identical to those described under Section 3.9. To summarize, we use the Gauss-Markov (G-M) source with correlation coefficient of 0.9, the AWGN channel (DMC equivalent used for hard decoding), and the SNR performance measure. Our experimental results are based on LT-FSVQ, whose next-state rule was obtained by omniscient design approach described in [49]. As mentioned earlier, the next-state rule is purely based on source statistics and is not optimized for the channel.

We first investigate the effectiveness of the IA, obtained by the optimization procedure suggested in Section 4.8. In order to do so, we compare in Fig. 4.6, the performance of a given ordinary FSVQ (designed for a noise-free channel) on a noisy channel with random IA (curve a) and optimized IA (curve b). Theses results were

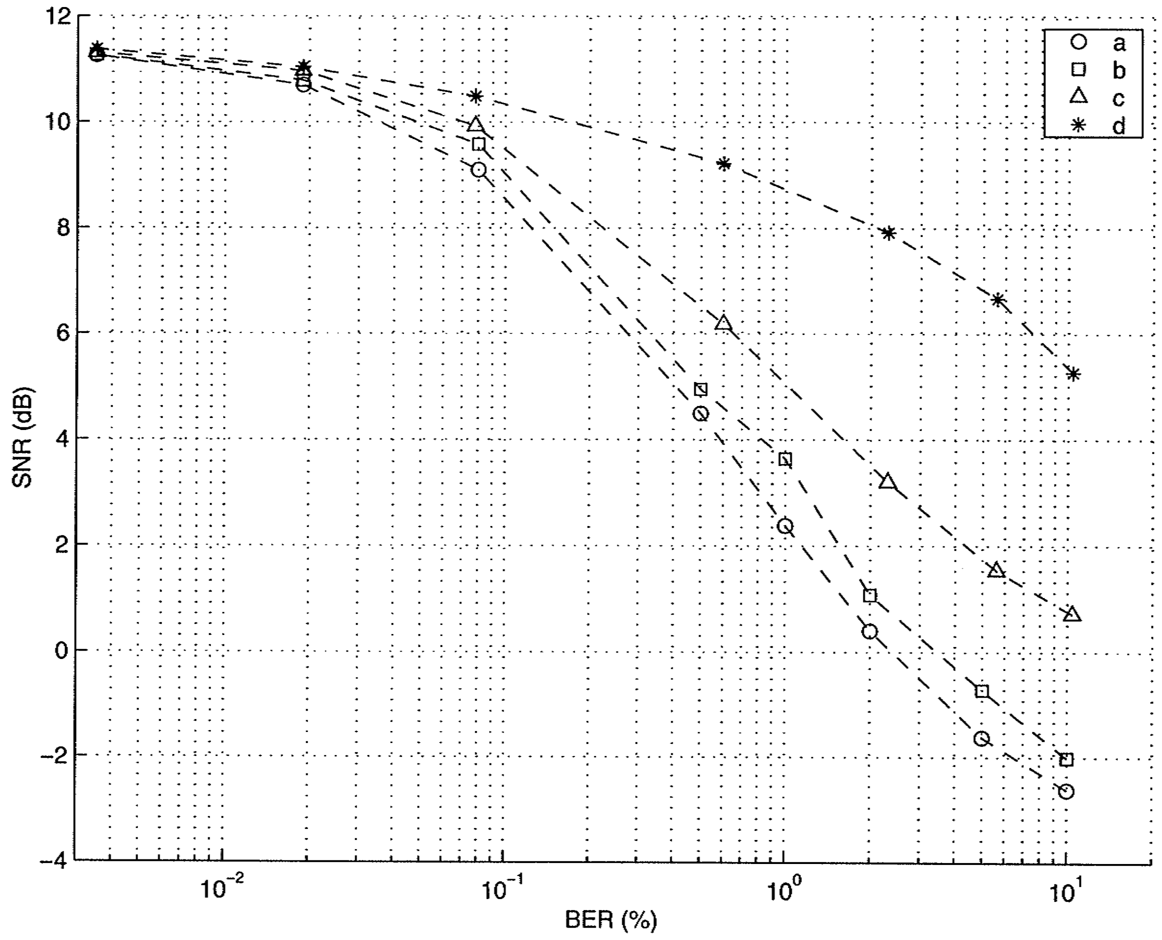


Figure 4.6: *Performance improvements due to proposed FSVQ design approaches compared to an ordinary FSVQ on Gauss-Markov source and AWGN channel. (a) FSVQ designed using OLT-FSVQ algorithm for noise-free channel, (b) FSVQ in (a) with IA optimized to channel noise-level, (c) FSVQ in (a) with proposed decoding algorithm (hard-decoding), and (d) channel optimized FSVQ (with hard-decoding) obtained with proposed iterative algorithm, using system in (a) as the initial system. In this example* $N = 16$, $d = 4$, *and* $K = 8$.

obtained with a 4-dimensional, 8-state FSVQ. We note that in this example, proper IA improves the performance on the average by about 0.8 dB in SNR at higher channel bit error rates (BER). In Fig. 4.6 curve c indicates the performance of the proposed decoding algorithm (with hard-decoding) with the same FSVQ encoder. Finally curve d shows the performance of COFSVQ (with hard-decoding) obtained by iteratively improving the given FSVQ to channel noise level using the algorithm in Table 4.2.

In our main experiments, 4-dimensional FSVQs with 8 states ($d = 4, K = 8$) were designed for G-M source and AWGN channels with varying levels of noise. In the case of hard decoding, a binary DMC was obtained after MAP detection, as described in Section 3.9. The designs were carried out using a training set of 50,000 vectors, while the testing was based on a separate set of 50,000 vectors from the same source. In order to average over channel noise density, 50 realizations of the channel noise sequence was used in soft-decoding experiments. The dramatic improvement in performance achieved by the proposed FSVQ designs over ordinary FSVQs on noisy channels is demonstrated by the example shown in Fig. 4.6.

Comparisons of performance of proposed channel optimized FSVQ (COFSVQ) designs with that of memoryless COVQ are shown in Figs. 4.7, 4.8, and 4.9. Some of these plots also include performance of memoryless COVQ at the same rate and dimension for comparison. Also included in some plots are the performance of NC-FSVQ2 designs (with 8 states) reported in [20] [1]. In these plots, the channel noise level is indicated in terms of CSNR defined in (3.42) and BER defined in (3.41). In general, COFSVQ appears to outperform memoryless COVQ in all cases considered. It is interesting to note that at low channel noise levels COFSVQ degrades much rapidly than COVQ. However, as the noise level is increased COFSVQ degrades slower than COVQ, and at very high noise levels COFSVQ maintains a gain of about 0.8-1.0 dB in SNR over COVQ. Comparisons with NC-FSVQ2 of [20] in Figs. 4.8 and 4.9 show that NC-FSVQ2 method performs better than COFSVQ with hard-decoding (NC-FSVQ2 is a hard-decoding based method) at high CSNRs, even though the situation appears to reverse as the noise level increases. Note also that COFSVQ with soft-decoding

[1] These values were approximately read-off the graphs shown in [20]; actual numerical values have not been reported.

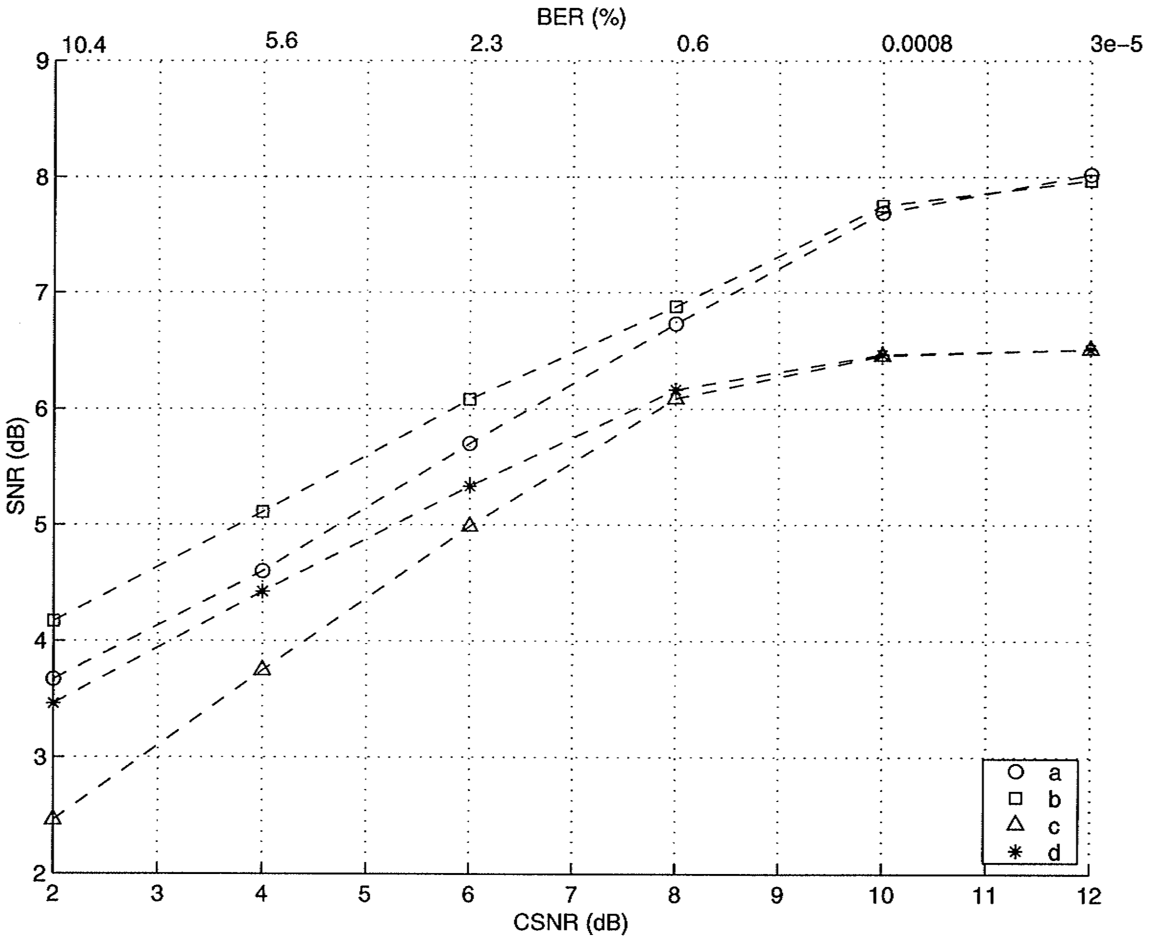


Figure 4.7: *Performance comparison of various noisy channel FSVQ schemes for Gauss-Markov source ($\rho = 0.9$) at 2 bits/vector, $d = 4$ and $K = 8$ (8-states): (a) proposed algorithm (hard-decoding), (b) proposed algorithm (soft-decoding), (c) memoryless COVQ with hard-decoding, (d) and memoryless COVQ with soft-decoding.*

outperforms NC-FSVQ2, particularly at high noise levels.

4.11 Summary

In this chapter, the problem of designing FSVQs for noisy channels was studied. We have proposed a robust, time-recursive decoder for optimally reconstructing the output of an FSVQ encoder, observed through a noisy channel. Experimental re-

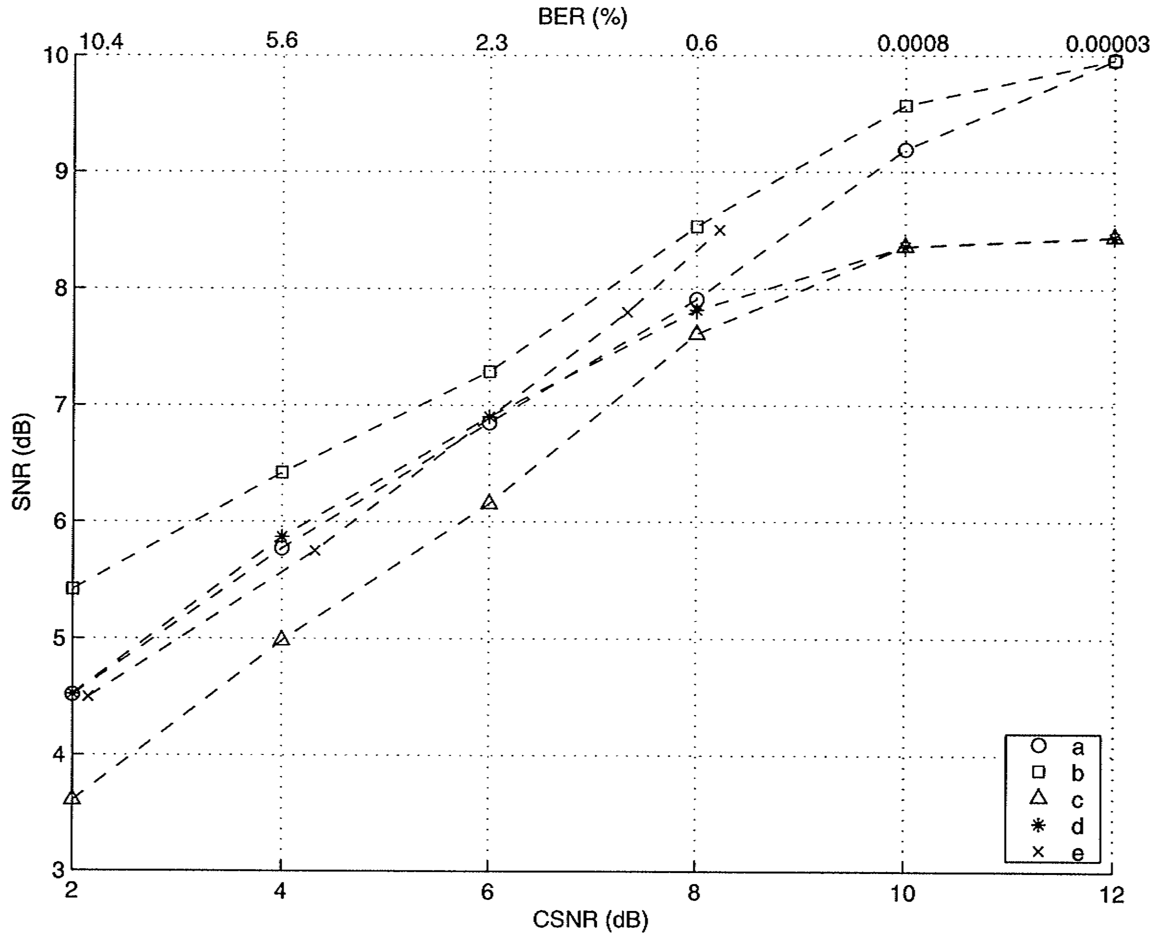


Figure 4.8: *Performance comparison of various noisy channel FSVQ schemes for Gauss-Markov source (ρ = 0.9) at 3 bits/vector, d = 4 and K = 8 (8-states): (a) proposed algorithm (hard-decoding), (b) proposed algorithm (soft-decoding), (c) memoryless COVQ with hard-decoding, (d)memoryless COVQ with soft-decoding, and (e) NC-FSVQ2 results from [20].*

sults were used to demonstrate the effectiveness of the proposed decoder. In contrast to a finite-state decoder, the proposed decoder exhibits graceful degradation of performance with increasing channel noise. The algorithm was also extended to soft-decoding. We also considered the iterative optimization of encoder and decoder for designing channel optimized FSVQ. Additionally, we derived a simulated-annealing based procedure for obtaining a good index assignment for state codebooks, which can

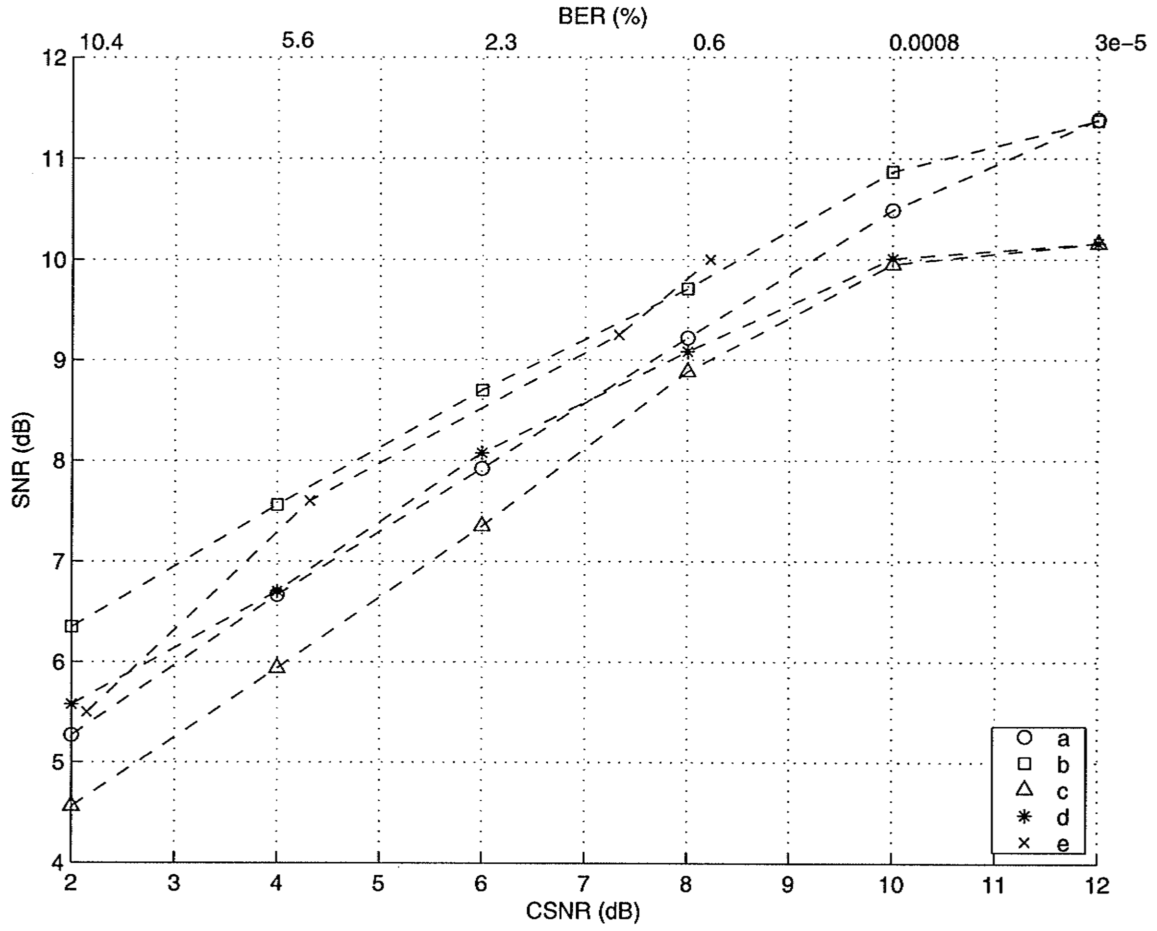


Figure 4.9: *Performance comparison of various noisy channel FSVQ schemes for Gauss-Markov source (ρ = 0.9) at 4 bits/vector d = 4 and K = 8 (8-states): (a) proposed algorithm (hard-decoding), (b) proposed algorithm (soft-decoding), (c) memoryless COVQ with hard-decoding, (d) memoryless COVQ with soft-decoding, and (e) NC-FSVQ2 results from [20].*

be useful in initializing the iterative design algorithm. Simulation results based on a Gauss-Markov source and the AWGN channel were presented and it was shown that robust FSVQ designed by methodology introduced in this chapter can outperform memoryless COVQ operating at the same rate.

Chapter 5

Soft-decoding VQ for Channels with Memory

5.1 Problem Statement And Motivation

In the VQ design problem introduced in the Chapter 2, a simple additive noise model was assumed for the channel. In this chapter, we consider the design of VQ for a general class of channels characterized by the model

$$\mathbf{y}_n = f(\mathbf{s}_n, \mathbf{s}_{n-1}, \ldots, \mathbf{s}_{n-M}) + \mathbf{w}_n, \tag{5.1}$$

where $M \geq 0$ is called the *channel memory*, f is a deterministic mapping such that $f : \mathbb{R}^{(M+1)L} \rightarrow \mathbb{R}^L$ and $\mathbf{w}_n \in \mathbb{R}^L$ is additive channel noise. According to this model, the channel output at a given symbol interval is "interfered" by M previous inputs to the channel, a phenomenon commonly referred to as *intersymbol interference* (ISI). A main cause of such interference is the limited bandwidth of the physical channel which causes the signal transmitted during one symbol interval to spread in time over several symbol intervals. In addition, ISI can also be caused by non-linearities in a communication system. For example, digital satellite systems often utilize amplifiers operating at or near saturation for better efficiency [64]. A general model for bandpass, non-linear channels is the *Volterra series representation* given by [65], [64].

$$\begin{aligned} \mathbf{y}_n &= \sum_{n_1} \mathbf{s}_{n-n_1} H^{(1)}_{n_1} + \sum_{n_1}\sum_{n_2}\sum_{n_3} \mathbf{s}_{n-n_1}\mathbf{s}_{n-n_2}\mathbf{s}^*_{n-n_3} H^{(3)}_{n_1 n_2 n_3} \\ &+ \sum_{n_1}\sum_{n_2}\cdots\sum_{n_5} \mathbf{s}_{n-n_1}\mathbf{s}_{n-n_2}\mathbf{s}_{n-n_3}\mathbf{s}^*_{n-n_4}\mathbf{s}^*_{n-n_5} H^{(5)}_{n_1 n_2 \ldots n_5} + \ldots + \mathbf{w}_n, \end{aligned} \tag{5.2}$$

where $H^{(2k-1)}_{n_1 n_2 \ldots n_{2k-1}}$ are the complex *Volterra coefficients.* We note that the first term represents the linear distortion, the second term the third order distortion and so forth for higher order distortion (even order terms are ignored as they generate spectral harmonics outside the channel bandwidth). A special case of this channel is the linear channel in which all coefficients except $H^{(1)}_{n_1}$ are zero.

In digital communications, a variety of methods exist for dealing with ISI. These methods are commonly known as *channel equalization*, the objective of which is to obtain an equivalent memoryless channel by appropriately processing a sequence of outputs from an ISI channel. If the equalizer provides discrete outputs, a hard-decoding COVQ can be designed for the equalized channel, using the procedure described in Chapter 2. However, it is well known that a considerable performance improvement may be achieved by using soft VQ decoders [25], [32]. The soft-decoding problem for channels with memory can be considered as a generalization of the MMSE channel equalization problem. A linear channel equalizer is a linear filter whose coefficients are found by MMSE estimation techniques [62]. Other work on MMSE equalization using non-linear filtering may be found in [65], [66], [67]. As we shall see, a soft VQ decoder may be considered as a non-linear time-invariant filter.

5.2 Optimal Soft-decoding

Referring to Fig. 2.2, the sequence of source vectors $\{\mathbf{X}_n\}$ is mapped to the channel vector sequence $\{\mathbf{S}_n\}$, which is then observed through a noisy channel. The overall

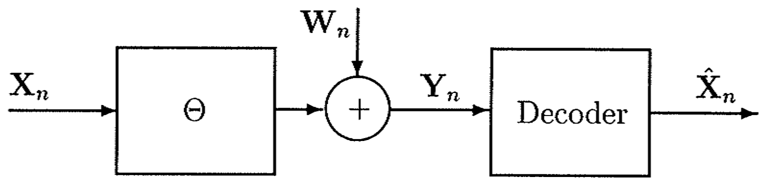


Figure 5.1: *Soft-decoding viewed as an estimation problem.*

mapping from the encoder input to the channel output is a non-linear mapping with memory, which can be written as (see Fig. 5.1)

$$\mathbf{y}_n = \Theta(\mathbf{x}_n, \ldots, \mathbf{x}_{n-M}) + \mathbf{w}_n. \tag{5.3}$$

Let $\bar{Y} = \{\mathbf{y}_n\}$ denote the observed channel output sequence, whose length can in theory be infinite. Also, let $p(\mathbf{x}_n, \bar{Y})$ be the joint density function between source vector $\mathbf{X}_n$ and $\bar{Y}$. If the decoder produces its output $\hat{\mathbf{x}}_n$ after observing $\bar{Y}$, the mean square error of the system is given by

$$\int_{\mathbb{R}^d}\int_{\mathbb{R}^d} \|\mathbf{x}_n - \hat{\mathbf{x}}_n(\bar{Y})\|^2 p(\mathbf{x}_n, \bar{Y}) d\mathbf{x}_n d\hat{\mathbf{x}}_n. \tag{5.4}$$

The optimal soft-decoder is a function $\hat{\mathbf{x}}_n^*(\bar{Y})$ which minimizes this error. The determination of the optimal decoder is an MMSE estimation problem, the solution to which is the conditional mean estimator (See Appendix A)

$$\hat{\mathbf{x}}_n^* = E\{\mathbf{X}_n|\bar{Y}\}. \tag{5.5}$$

Clearly, if such a decoder is to be of practical interest, $\bar{Y}$ must be a truncated sequence. Apart from restrictions imposed by obvious computational difficulties, in most applications (*e.g.*, speech or image coding for on-line communication), there are restrictions on the allowable delay. Hence we focus on optimal decoding subject to a constraint on decoding delay.

A typical approach to dealing with the estimation problem in (5.5) is to use a sequence of the form $\bar{Y}_n = (\mathbf{y}_1, \ldots, \mathbf{y}_n, \ldots, \mathbf{y}_{n+k})$, where k is the decoding delay. In the terminology of estimation theory, this problem in is referred to as *filtering* if $k = 0$ and *smoothing* if $k > 0$ [51]. In general smoothing results in more accurate estimates than filtering as more observations closer to the time point n are used in the former than the latter to estimate $\mathbf{X}_n$. However, this gain can only be obtained at the cost of increased decoding delay and complexity. When a constant delay k is used for all n, the resulting smoother is referred to as a *fixed-lag smoother* [51]. In general the formulation of the problem in this manner leads to a recursive solution. For example if the mapping from $\mathbf{X}_n$ to $\mathbf{Y}_n$ is linear and if the two process are jointly Gaussian, the problem can be solved recursively using the *Kalman filter* [51]. This approach is not applicable to the system that we consider, due to the non-linear mapping involved in the VQ encoder. However, the fixed-lag smoother can be implemented in a recursive manner in the context of soft VQ decoding as well. The approach has previously been investigated in [31], [32], [33] and we consider this approach in the next section. In the rest of the chapter, we investigate a *sliding-block smoothing* solution to the soft-decoding problem [68], [69]. A useful property of the resulting decoder is that it is a time-invariant filter, which allows us to estimate the optimal decoder using training data, much the same way MMSE channel equalizers are estimated. Also we will show that the optimal sliding-block decoder can be well approximated by a linear filter when the CSNR is small.

5.3 Recursive Soft-decoding

Consider the soft-decoder of the form

$$\begin{aligned}\hat{\mathbf{x}}_n^* &= E\{\mathbf{X}_n|\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{n+k}\} \\ &= \sum_{\bar{\mathbf{s}}_{n+k}} E\{\mathbf{X}_n|\bar{\mathbf{s}}_{n+k}, \bar{Y}_n\} P(\bar{\mathbf{s}}_{n+k}|\bar{Y}_n) \\ &= \sum_{\bar{\mathbf{s}}_{n+k}} E\{\mathbf{X}_n|\bar{\mathbf{s}}_{n+k}\} P(\bar{\mathbf{s}}_{n+k}|\bar{Y}_n). \qquad (5.6)\end{aligned}$$

where $\bar{\mathbf{S}}_{n+k} = (\mathbf{S}_{n-1+M}, \ldots, \mathbf{S}_{n+k})^T$. In the above expression, the terms $E\{\mathbf{X}_n|\bar{\mathbf{s}}_{n+k}\}$ depend only on source statistics and the encoder, while the posterior probabilities $P(\bar{\mathbf{s}}_{n+k}|\bar{Y}_n)$ depend on the observed channel outputs. As the number of terms in the sum is a function of time, the evaluation of this expression is clearly impractical. However, if we make some assumptions about the encoder output $\{\mathbf{S}_n\}$, it is possible to obtain an expression that lends itself to recursive computation. Consider for example that, if $\{\mathbf{S}_n\}$ is assumed to be an iid process, (5.6) simplifies to

$$E\{\mathbf{X}_n|\bar{Y}_n\} = \sum_{i=1}^{N} E\{\mathbf{X}_n|\alpha_i\} P(\alpha_i|\bar{Y}_n), \qquad (5.7)$$

where $E\{\mathbf{X}_n|\alpha_i\}$ $i = 1, \ldots, N$ are simply the encoder centroids. A more useful model for the encoder output process is the first-order Markov model. Specifically, if we assume that both $\{\mathbf{X}_n\}$ and $\{\mathbf{S}_n\}$ are Markov the following assumption is reasonable

[32]:

$$E\{\mathbf{X}_n|\mathbf{s}_{-M+1},\ldots,\mathbf{s}_{n+k}\} \approx E\{\mathbf{X}_n|\mathbf{s}_{n-1},\ldots,\mathbf{s}_{n+k}\}. \tag{5.8}$$

Then, the expression in (5.6) reduces to

$$E\{\mathbf{X}_n|\bar{Y}_n\} = \sum_{i=1}^{N^{k+2}} E\{\mathbf{X}_n|\mathbf{v}_n^{(i)}\}P(\mathbf{v}_n^{(i)}|\bar{Y}_n), \tag{5.9}$$

where $\mathbf{V}_n = (\mathbf{S}_{n-1},\ldots,\mathbf{S}_{n+k})^T$ and $\mathbf{v}_n^{(i)}$ denote the i^{th} permutation of $(k+2)$-tuple of N-ary vectors $\mathbf{S} \in \{\alpha_1,\ldots,\alpha_N\},$, $i = 1,\ldots,N^{k+2}$. The *residual redundancy* [52] in the channel input due to correlation in the encoder output can be utilized as protection against channel distortion at the receiver by an appropriate design of the decoder. The iid assumption made in (5.7) clearly ignores the residual redundancy in the encoder output. In (5.9) vectors $E\{\mathbf{X}_n|\mathbf{v}_n^{(i)}\}$, $i = 1,\ldots,N^{k+2}$ define a set of centroids based on the index sequences $\mathbf{v}_n^{(i)}$, which we will refer to as *extended centroids*[1]. Clearly (5.7) is a special case of (5.9); the latter reduces to former if the encoder output is iid. If the encoder output is correlated, a block of output indices $\mathbf{V}_n$ defines a higher resolution partition of $\mathbb{R}^d$ than a single output index $\mathbf{S}_n$, and the decoder in (5.9) provides improved MSE performance compared to that of (5.7). However, as shown below, this improvement comes at the cost of increased computational complexity.

In (5.9), the only quantities which need be evaluated for every channel output are the posterior probabilities $P(\mathbf{v}_n^{(i)}|\bar{Y}_n)$. It can be shown that $P(\mathbf{v}_n^{(i)}|\bar{Y}_n)$, $i = 1,2,\ldots$ can be updated recursively from $P(\mathbf{v}_{n-1}^{(i)}|\bar{Y}_{n-1})$, $i = 1,2,\ldots$ [50], [32]. This leads to a recursive algorithm for soft decoding, which may be viewed as an extension of the symbol-by-symbol MAP channel equalization algorithm of Abend and Fritchman [50]. We note here that this algorithm requires the knowledge of the conditional density of

[1]Skoglund [32] refers to these as multi-centroids.

the channel output, given the channel inputs. Simulation results reported in [32] based on this approach show that for channels with severe ISI, soft-decoding can provide a significant performance improvement over hard decoding. A major drawback of the recursive soft-decoding algorithm is that, it's computational complexity grows as $O(N^{k+3})$ for $k \geq M$ ($O(N^{M+2})$ for $k \leq M$). Note that the decoding delay k has to be increased with channel memory M. Hence, this decoder becomes impractical for channels with larger memory or quantizers with higher rates.

5.4 Sliding-block Decoding

When the input process and the channel have finite memory, it can be assumed that the dependence of $\mathbf{X}_n$ on $\mathbf{Y}_m$ decreases as $|n-m|$ is increased. More specifically, we will assume that

$$p(\mathbf{x}_n, \mathbf{y}_1, \ldots, \mathbf{y}_n, \mathbf{y}_{n+1}, \ldots) \approx p(\mathbf{x}_n, \mathbf{y}_{n-K_1}, \ldots, \mathbf{y}_n, \ldots, \mathbf{y}_{n+K_2}), \tag{5.10}$$

for $n \gg 1$, where K_1 and K_2 are some positive integers. This leads to the sliding-block smoother (or sliding-window smoother) given by

$$\hat{\mathbf{x}}_n = E\{\mathbf{X}_n | \mathbf{y}_{n-K_1}, \ldots, \mathbf{y}_n, \ldots, \mathbf{y}_{n+K_2}\}. \tag{5.11}$$

Given that source and channel are stationary, the sliding-block smoother is a time-invariant (fixed), non-linear function $\Phi(.)$ of the observed vector

$$\mathbf{U}_n = (\mathbf{Y}_{n-K_1}, \ldots, \mathbf{Y}_n, \ldots, \mathbf{Y}_{n+K_2})^T,$$

where $\mathbf{Y}_n \in \mathbb{R}^L$. That is

$$\Phi : \mathbb{R}^{(K_1+K_2+1)L} \to \mathbb{R}^d. \tag{5.12}$$

We will refer to the soft-decoder based on (5.11) as the *sliding-block decoder.* It is illustrated in Fig 5.2. Note that K_2 is the decoding delay. The "block-size" in channel vectors is $K_0 = K_1 + K_2 + 1$, which is also the total memory of the decoder.

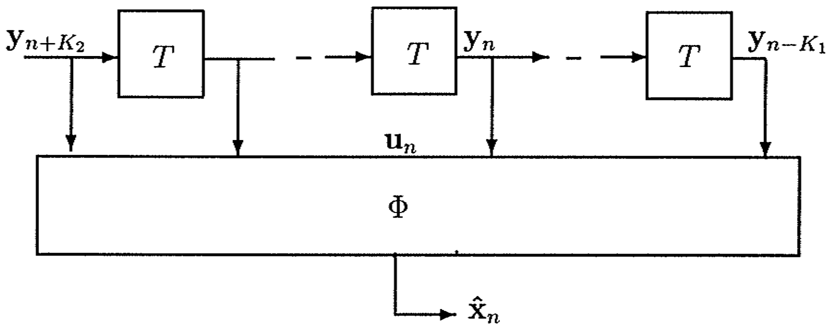


Figure 5.2: *Sliding block decoder (T is a unit delay element).*

Let $\mathbf{V}_n = (\mathbf{S}_{n-K_1-M}, \ldots, \mathbf{S}_{n+K_2})^T$. Then, noting that the sub-sequence of channel outputs $\mathbf{U}_n$ conditionally depends only on the sub-sequence of channel inputs $\mathbf{V}_n$, we can express the sliding-block decoder as,

$$\begin{aligned}
\phi(\mathbf{u}_n) &= E\{\mathbf{X}_n|\mathbf{u}_n\} \\
&= \sum_i E\{\mathbf{X}_n|\mathbf{v}_n^{(i)}, \mathbf{u}_n\}P(\mathbf{v}_n^{(i)}|\mathbf{u}_n) \\
&= \frac{\sum_i E\{\mathbf{X}_n|\mathbf{v}_n^{(i)}\}p(\mathbf{u}_n|\mathbf{v}_n^{(i)})P(\mathbf{v}_n^{(i)})}{\sum_i p(\mathbf{u}_n|\mathbf{v}_n^{(i)})P(\mathbf{v}_n^{(i)})} \\
&= \frac{\sum_i \mathbf{g}'_i p(\mathbf{u}_n|\mathbf{v}_n^{(i)})P(\mathbf{v}_n^{(i)})}{\sum_i p(\mathbf{u}_n|\mathbf{v}_n^{(i)})P(\mathbf{v}_n^{(i)})}
\end{aligned} \tag{5.13}$$

where $\mathbf{g}'_i = E\{\mathbf{X}_n|\mathbf{v}_n^{(i)}\}$ and $i = 1, \ldots, N^{K_0+M}$. In (5.13), the only term that depends on the channel is the conditional density $p(\mathbf{u}_n|\mathbf{v}_n^{(i)})$, which can be computed from the channel noise density, if channel input and noise are assumed independent. On the other hand, the set $\mathbb{G} = \{\mathbf{g}'_i\}$ in this case defines a set of extended centroids of the encoder, whose size depends on K_0. That is, they are centroids of the partition of $\mathbb{R}^d$ based on $K_0 + M$ consecutive outputs from the encoder. When the encoder output is correlated (there is residual redundancy), the cardinality of $\mathbb{G}$ is larger than N. It is apparent that the optimal sliding-block decoder given by (5.13) simply computes the conditional expectation of a set of extended centroids, given the augmented output vector $\mathbf{u}_n$. Hence the advantages of the sliding block-decoder are two-fold: (i) it uses residual redundancy (if any) by using the knowledge of an extended set of centroids, and (ii) it compensates for channel spread by using an augmented observation vector. The performance of the sliding-block decoder depends on its memory span, determined by the block-size K_0. On one hand, increasing K_0 can be expected to improve the MSE performance of the decoder, albeit increasing the complexity of the decoder mapping ϕ. On the other hand, the finite memory properties of the source and channel suggest that a finite, and possibly a small value of K_0 may be practically sufficient to obtain all the improvements achievable with having memory in the decoder. Hence, we wish to investigate the dependence of overall MSE performance on the block-size K_0. At this point, it may be conjectured that the optimal value of K_0 should depend on channel characteristics and the amount of correlation in channel input process.

Example

We present here numerical results obtained by applying sliding-block decoding to the output of a VQ encoder transmitted over a linear Gaussian channel. We consider two sources; Gauss-Markov (G-M) source with a correlation coefficient of 0.9 (see Appendix C) and iid Gaussian source. The former is a highly correlated source and results in a considerable encoder residual redundancy, while the latter leaves very little residual redundancy. Here, we restrict our attention to two-dimensional vector quantization ($d = 2$) at the channel rate of 1 bit per source vector ($N = 2$) and binary channel signaling. That is, we have a binary quantizer operating at the rate of 1 channel use per source vector. This situation allows us to analytically compute the decoder output and hence measure the performance of the sliding-block decoder, without resorting to function approximation. The binary channel has the impulse response $h_0 = 0.407, h_1 = 0.815, h_2 = 0.407$ and a memory of $M = 2$ (See (5.22)). The performance of sliding-block decoding was evaluated by simulating the source and the channel. The SNR of the quantizer with decoding block-size K_0 bits[2] for different channel noise levels and for different values of K_0 is shown in Figs 5.3 and 5.4. The channel noise level is measured here by the channel signal-to-noise ratio (CSNR) defined as

$$10 \log_{10} \left(\frac{h_0^2 + h_1^2 + h_2^2}{\sigma_W^2} \right) \text{ dB.} \tag{5.14}$$

Clearly, the largest improvement in performance is achieved by increasing the K_0 from 1 to 3 bits in both figures. It is also noticeable that the relative gain in performance achieved beyond $K_0 = 5$ bits is small. Interestingly, this value of K_0 corresponds to $K_1 = K_2 = 2$ bits, which is also the memory of the channel. In the case of iid source, none of the decoders are able to achieve a performance close to ideal channel performance on the given channel, except at very high CSNRs. In contrast, in the case of G-S source, all decoders with non-zero memory achieve a performance superior to that of an ideal channel quantizer at CSNR > 5 dB. This is due to the high residual redundancy in the encoder output in the latter case. Fig. 5.5 shows the extended sets of encoder centroids used by a sliding-block decoder of block-size K_0,

[2]For simplicity, symmetric blocks have been used, so that $K_1 = K_2$.

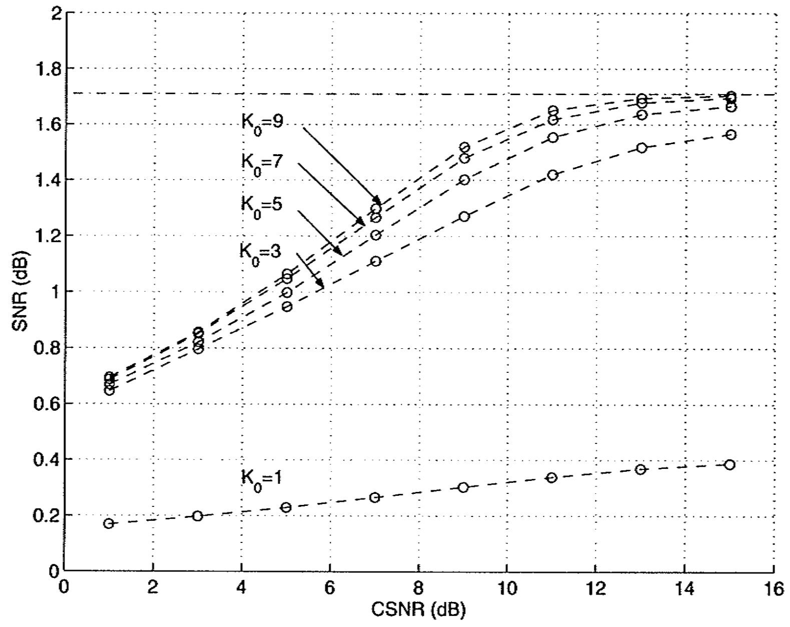


Figure 5.3: *Dependence of performance on decoder block-size K_0 (bits) for iid Gaussian source, $N = 2$, and $d = 2$. Channel memory $M = 2$ bits. Distortion of quantizer over ideal channel is* 1.71 *dB (horizontal line). In each case a training set of 75000 source vectors was used.*

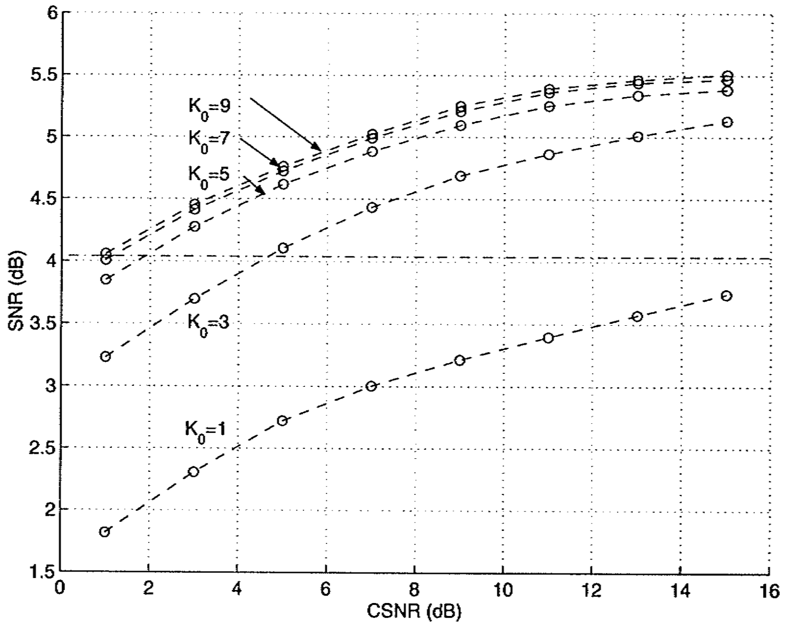


Figure 5.4: *Dependence of performance on decoder block-size K_0 (bits) for G-M source, $N = 2$, and $d = 2$. Channel memory $M = 2$ bits. Distortion of quantizer over ideal channel is* 4.04 *dB (horizontal line). In each case a training set of 75000 source vectors was used.*

with iid and G-S sources respectively. With the iid source, extended centroids almost coincide with the 2 centroids of the encoder partition and no residual redundancy is thus present. In contrast, high residual redundancy is evident in the case of correlated G-S source, where extended centroids are spread across the support-region of input vectors.

5.5 Approximations for Low CSNR and Gaussian Noise

The optimal sliding-block decoder given by (5.13) is a non-linear function. However, a result shown in [70] for the case of the memoryless AWGN channel motivates us to consider linear approximations for sliding-block decoder at low CSNRs. It is shown in [70] that as CSNR $\to 0$, the optimal non-linear decoder for a memoryless Gaussian channel tends to a linear mapping. As the optimal decoder for memoryless channel is a simple case of sliding-block decoding (in which the block size is 1 channel vector), the result in [70] can also be shown to be valid for a sliding-block decoder considered here. More precisely we can show that (derivation in Appendix F)

$$\phi(\mathbf{u}_n) = G\mathbf{u}_n + o(\sqrt{E_0}, \mathbf{u}_n) \quad \text{as } E_0 \to 0, \tag{5.15}$$

where G is a $d \times K_0 L$ matrix that is fixed for a given source, encoder, and a channel (see F.8), and E_0 is the average signal power at the channel output. According to this result it appears that the optimal sliding block decoder approximates a linear function at low CSNRs. From a practical view point, this implies that the optimal decoder mapping can be well approximated by a simpler function, when CSNR is low. Our experimental results seem to support this conjecture.

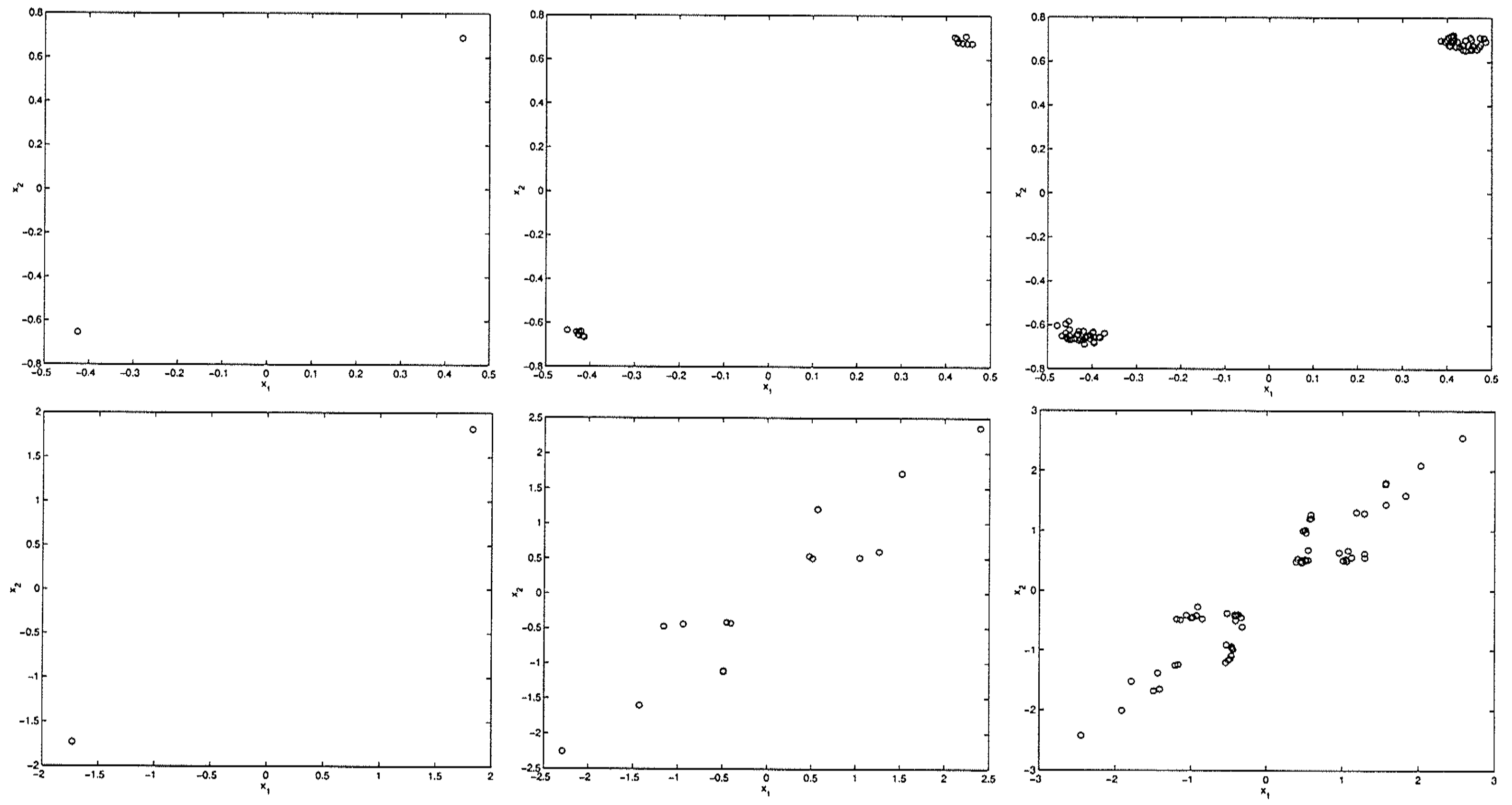


Figure 5.5: *Extended sets of encoder centroids used by sliding-block decoder for iid source (top row) and Gauss-Markov source (bottom row). Encoder has $N = 2$ and channel memory $M = 2$.* **left**: *the 2 centroids of decoder partition,* **middle**: $K_0 = 2$, *(16 extended centroids), and* **right**: $K_0 = 4$, *(64 extended centroids).*

5.6 Estimation of Sliding-block Decoder

Consider the optimal decoder in (5.13), when the channel noise is iid Gaussian with covariance matrix diag(σ_W^2). It directly follows that

$$p(\mathbf{u}_n|\mathbf{v}_n^{(i)}) = \prod_{j=0}^{K_1+K_2} p_W(\mathbf{y}_{n+K_2-j} - \mathbf{r}_{n+K_2-j}^{(i)}), \tag{5.16}$$

where $\mathbf{r}_{n+K_2-j}^{(i)} = f(\mathbf{s}_{n+K_2-j}^{(i)}, \ldots, \mathbf{s}_{n+K_2-j-M}^{(i)})$, with f being defined in (5.1). After some work, it can be shown that

$$E\{\mathbf{X}_n|\mathbf{u}_n\} = \frac{\sum_{i=1}^{N^{K_0+M}} \mathbf{g}_i' P(\mathbf{v}_n^{(i)}) \exp\Big(-\frac{1}{2\sigma_W^2}\sum_{j=0}^{K_1+K_2} \|\mathbf{y}_{n+K_2-j} - \mathbf{r}_{n+K_2-j}^{(i)}\|^2\Big)}{\sum_{i=1}^{N^{K_0+M}} P(\mathbf{v}_n^{(i)}) \exp\Big(-\frac{1}{2\sigma_W^2}\sum_{j=0}^{K_1+K_2} \|\mathbf{y}_{n+K_2-j} - \mathbf{r}_{n+K_2-j}^{(i)}\|^2\Big)} \tag{5.17}$$

The number of terms N^{K_0+M} in each sum above can be quite high even for small values of N, M, and K_0. However, some simplifications may be possible in the case of highly correlated sources, as not all permutations of $\mathbf{V}_n$ are likely to occur, *i.e.*, $P(\mathbf{v}_n^{(i)}) \approx 0$ for some i. Even so, for very large values of N^{K_0+M}, enumerating through all the possibilities can be impossible. We suggest below an alternative approach based on approximation of the expression in (5.13) by a non-linear function, which can be estimated from training data. It is then a problem of *regression estimation.* This approach can also be viewed as an extension of MMSE channel equalization using linear or non-linear filters. The rest of this chapter is devoted to investigating the feasibility of this approach. Before we proceed, it is worth mentioning that the estimation of sliding-block decoder can be used with a much larger class of channels than the one described by (5.1). We only require that the input vector $\mathbf{X}_n$ and the vector sequence $\mathbf{U}_n$ observed at the channel output have a stationary joint density $p(\mathbf{x}_n, \mathbf{u}_n)$.

Let $\{\mathbf{x}_i^{(t)}\}_{i=1}^{n_T}$ be a sequence of training vectors representing the source distribution. Given the encoder and channel, one can compute n_T realizations $\{\mathbf{u}_i^{(t)}\}_{i=1}^{n_T}$ of the channel output sequence. Our objective is to estimate a function $\hat{\Phi}$ by minimizing the empirical error

$$\frac{1}{n_T}\sum_{i=1}^{n_T} ||\mathbf{x}_i^{(t)} - \hat{\Phi}(\mathbf{u}_i^{(t)})||^2. \tag{5.18}$$

There exists a large number of methods for function estimation. However, many of these methods suffer from "the curse of dimensionality" in the multi-dimensional (sparse data) case, see for example [71]. Note that we have to estimate a d-dimensional function of K_0L variables. An empirical comparison of several function estimation methods can be found in [72]. In general, the accurate estimation of high-dimensional functions requires methods based on projection of high-dimensional data onto low-dimensional sub-spaces. In this thesis, we consider one such approach- *multi-layer perceptron* (MLP) . We indicate other possibilities in Section 6.2.

Multi-layer Perceptron

Multi-layer perceptron [72] is a neural-network approach that has been extensively studied for multi-dimensional function estimation and the related problem of classification. Applications of MLP related to our problem include adaptive channel equalization [66] and multi-user detection [73]. The universal approximation theorem [74] implies that a multi-layer perceptron possesses the universal function approximation capability in that, a multi-layer perceptron with a single hidden layer can approximate any continuously differentiable function to an arbitrary accuracy, provided that the hidden layer has a sufficient number of computing nodes. A single hidden layer MLP is a non-linear, continuous function in the unit hypercube $[0, 1]^p$

given by

$$f(u_1, u_2, \ldots, u_p) = \sum_{i=1}^{q} \rho_i \sigma(\sum_{j=1}^{p} w_{ij} u_j - \theta_i) - \rho_0 \tag{5.19}$$

where ρ_i, θ_j, and w_{ij} are some real constants, and $\sigma(.)$, usually referred to as the *activation function*, is a *sigmoidal function* -a real nondecreasing function such that $\sigma(t) \to -1$ as $t \to -\infty$ and $\sigma(t) \to 1$ as $t \to \infty$. An example is the logistic sigmoid $\sigma(t) = (1 - e^{-t})/(1 + e^{-t})$, which was used in our simulations. The function f corresponds to an MLP with p-dimensional inputs and q computing nodes in the hidden layer. Each computing node or a *neuron* in the hidden layer maps an input vector onto a real scalar and the output of each hidden node is then linearly combined in the output layer. This suggests that approximation in an MLP is performed after projecting input vectors onto a lower dimensional sub-space. While a single-hidden layer MLP is sufficient for approximating any continuous function to an arbitrary accuracy, the number of computing nodes required in the hidden layer may be very high. It has been found in practice that MLPs with two hidden layers result in simpler implementations and reduced learning times, for the same estimation error.

It is evident from (5.19) that MLP is a means of parameterizing a multi-dimensional mapping. In particular, the function $f(\mathbf{u}; \mathbf{w})$ in (5.19) is determined by the parameter vector $\mathbf{w} = (\rho_i, w_{ij}, \theta_j)$ where $i = 1, \ldots, q$ and $j = 1, \ldots, p$ (these parameters are also called *weights* in neural-network terminology). Given a training set $\{(\mathbf{x}_i^{(t)}, \mathbf{u}_i^{(t)})\}_{i=1}^{n_T}$, fitting of an MLP to a data set (network *training* or *learning*) involves choosing the parameter vector $\mathbf{w}$ which minimizes the error

$$J(\mathbf{w}) = \sum_i \|\mathbf{x}_i^{(t)} - f(\mathbf{u}_i^{(t)}; \mathbf{w})\|^2. \tag{5.20}$$

Clearly, this is not a quadratic function in $\mathbf{w}$ and therefore the minimization is commonly achieved through iterative descent algorithms. The basic estimation algorithm for MLP is the *back-propagation* [75], [76], to which many variants exist. In the simplest case, the weights are updated to reduce $J(\mathbf{w})$ in the direction of steepest descent as

$$\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla_{\mathbf{w}} J, \tag{5.21}$$

where η is the *learning rate.* In order to speed-up the convergence, various modifications to the basic algorithm can be used. While some of these are heuristic in nature (*e.g.*, momentum, variable rate, resilient back-propagation) others are based on standard numerical optimization methods (*e.g.*, conjugate-gradient, quasi-Newton, Levenberg-Marquardt), see [75], [76]. We do not discuss the relative merits of various training algorithms here, but refer to [77] for a detailed case study. For practical reasons, we used the resilient back-propagation algorithm RPROP [77] in all our simulations.

The computational complexity of an MLP decoder depends on the number of hidden nodes. Each hidden node requires the evaluation of an inner product and the value of a scalar function (sigmoid). Additionally, each output node requires an inner product. Thus, for an MLP with p inputs, d outputs, and 2 hidden layers with q nodes each, $p(q+1)+(q+1)^2+(q+1)d$ multiplications and additions are required to compute all the inner products. MLP decoders used in our simulations have much less computational complexity than (comparable (in performance) recursive soft-decoders. It is also worth pointing out that MLPs are considered very efficient computing architectures as they can be implemented in parallel. In any case, most of the computing effort in an MLP is required in the training phase. As we consider only off-line (not adaptive) estimation, the complexity of training does not affect the speed of the MLP-decoder.

5.7 Experimental Results

In this section, the performance of the sliding-block decoder is experimentally obtained for Gauss-Markov (G-M) source (described in Appendix C) and the linear Gaussian channel given by

$$\mathbf{y}_n = 0.407\mathbf{s}_n + 0.815\mathbf{s}_{n-1} + 0.407\mathbf{s}_{n-2} + \mathbf{w}_n, \tag{5.22}$$

where $\mathbf{s}_n = \{+1.0, -1.0\}$ (anti-podal signaling) and $\mathbf{w}_n$ is iid Gaussian noise . This channel has a spectral-null and hence exhibits severe ISI (linear channel equalization is ineffective) [62]. Previously, the performance of several other soft-decoding schemes on the same channel was reported in [32] and [30]. While the memory of this binary channel $M_b = 2$, the amount of channel memory induced by the equivalent L dimensional channel is $M = 1$ for $L > 1$. This is the case in all our simulations. We have considered two cases of G-M source: highly correlated ($\rho = 0.9$) and uncorrelated ($\rho = 0$). In the highly correlated case, a considerable encoder residual redundancy can be expected (especially for small vector dimensions) while in the uncorrelated case the residual redundancy is negligible [3]. The estimation of encoder and decoder parameters was carried out using a training set of source vectors while the performance of the resulting systems was evaluated using a separate set of source vectors.

In this section, the performance of the sliding-block decoder is also compared with the performance of both a hard-decoder and a recursive soft-decoder. A hard-decoding scheme for a channel with memory requires an equivalent discrete memoryless channel (DMC) model. In the scheme used in our simulations, a DMC was obtained by using a maximum-likelihood sequence detector, implemented by the Viterbi algorithm (VA)

[3]Here, by residual redundancy, we mean the redundancy due to memory in the process.

[62]. The transition matrix of the resulting DMC was estimated through simulations. Performance of VA depends on the decoding delay used; a larger delay gives better performance but also increases the storage requirements. A rule of thumb is to use a delay $\geq 5M_b$ (see page 561 of [62]). In the following we adopt the following notation for simplicity: a sliding-block soft-decoder with a memory (block-size) of K_0 bits and delay of K_2 bits is referred to as $SB - SD(K_2, K_0)$; a recursive soft-decoder using a delay of K_2 bits is referred to as $R - SD(K_2)$; a hard decoder based on VA with a K_2 -bit delay is simply referred to as $VA - HD(K_2)$.

In our simulations, the estimation of the MLP decoder is carried out using a training set of given size n_T (fixed). Due to practical reasons mentioned earlier, here we use networks with 2 hidden-layers. Hence, there are two parameters which determine the performance of the resulting decoder: decoder block-size K_0 and the number of hidden-nodes n_H per hidden layer (we simply use equal number of nodes in each hidden layer). As described before, the K_0 has to be selected to match the channel impulse response and the encoder statistics. In the context of estimation, selection of both of these parameters has other implications as well. In function estimation problems there exists two sources of error. First is the approximation error due to incapacity of the chosen model (MLP) to represent the true function implied by the observed data. Second is the estimation error due to randomness in the parameters estimated from a finite-size training set. A reduction in the approximation error requires an increase in the number of free parameters of the model (number of MLP weights determined by n_H), while a reduction in the estimation error requires a decrease in n_H (*i.e.*, increase in n_T/n_H). Note also that, increasing K_0 increases the number of free parameters in the model. Hence, there exists optimal values for K_0 and n_H for a fixed n_T. Tables 5.1, 5.2, and 5.3 show the dependence of empirical

MSE of the quantizer on K_0 (in terms of channel output vectors) and n_H, for fixed n_T. All MLP decoders have been estimated using the same channel input sequence (*i.e.*, the same encoder).

K_0	delay	SNR (dB)			
	(bits)	$n_H = 10$	$n_H = 15$	$n_H = 20$	$n_H = 25$
3	2	7.4	7.6	7.6	7.6
5	4	7.8	8.0	8.2	8.2
7	6	8.0	8.1	8.3	8.3
9	8	8.2	8.3	8.3	8.4

Table 5.1: *SNR of quantizers with sliding-block decoding over the channel given by (5.22)- CSNR=13 dB, $d = 2$, $N = 4$, and $n_T = 35000$. SNR of the ideal channel quantizer is 7.9 dB.*

K_0	delay	SNR (dB)			
	(bits)	$n_H = 10$	$n_H = 15$	$n_H = 20$	$n_H = 25$
3	3	9.4	9.6	9.8	9.8
5	6	9.5	9.6	9.8	9.8
7	9	9.6	9.7	9.9	9.9

Table 5.2: *SNR of quantizers with sliding-block decoding over the channel given by (5.22)- CSNR=13 dB, $d = 3$, $N = 8$, and $n_T = 40000$. SNR of the ideal channel quantizer is 9.4 dB.*

K_0	delay	SNR (dB)			
	(bits)	$n_H = 10$	$n_H = 15$	$n_H = 20$	$n_H = 25$
3	4	9.9	10.2	10.4	10.4
5	8	10.0	10.3	10.4	10.4
7	12	10.0	10.3	10.4	10.4

Table 5.3: *SNR of quantizers with sliding-block decoding over the channel given by (5.22)- CSNR=13, dB $d = 4$, $N = 16$, and $n_T = 50000$. SNR of the ideal channel quantizer is 10.1 dB.*

We next present a comparison of sliding-block decoder with recursive soft-decoder and Viterbi-based hard-decoder. As mentioned earlier, the practical implementation of the recursive decoder requires a statistical model for the encoder output. For

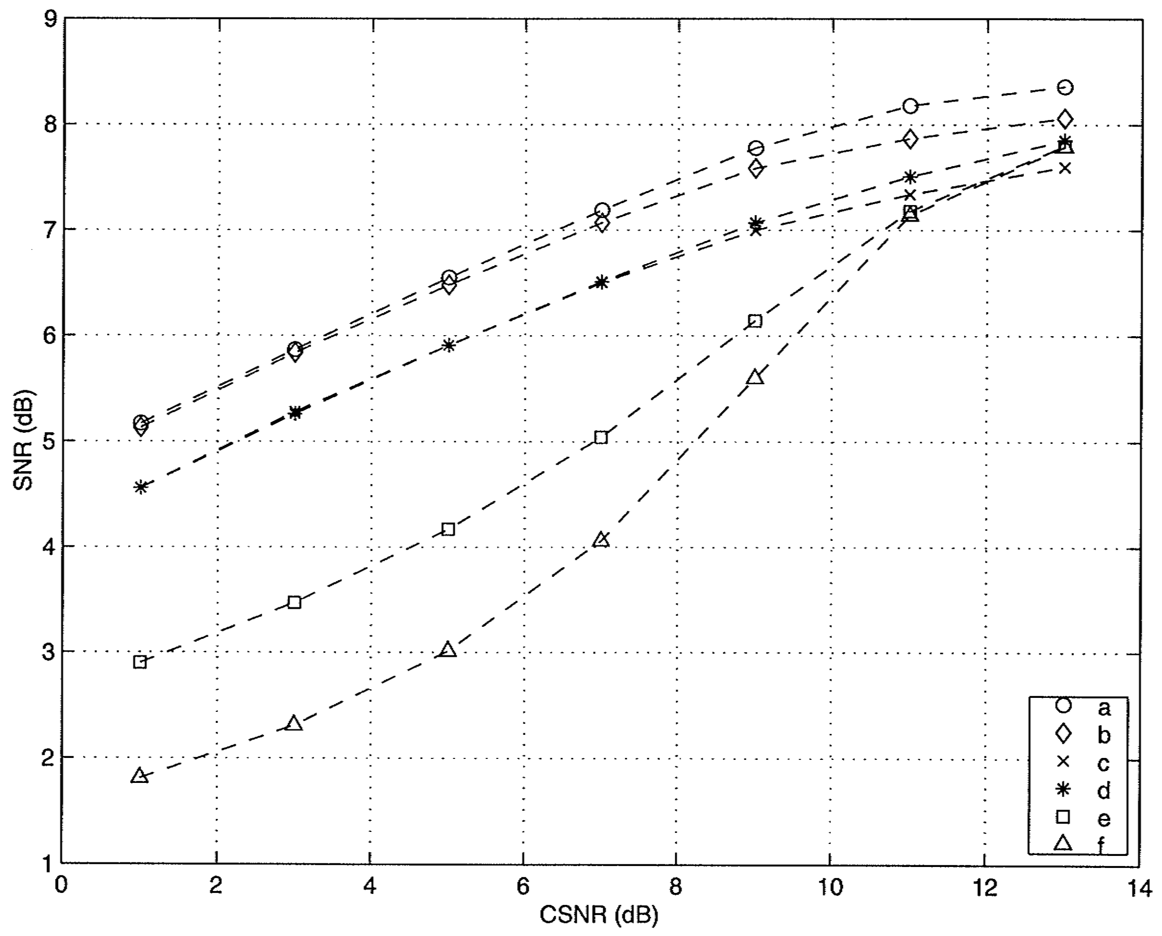


Figure 5.6: *Performance comparison of soft and hard decoding for $d = 2$, $N = 4$: (a) SB-SD(10,22) using MLP, (b) SB-SD(4,10) using MLP, (c) SB-SD(2,6) using MLP, (d) SB-SD(2,6) using the analytical equation (5.17), (e) R-SD(2), and (f) VA-HD(20). All MLP-based sliding-block decoders were 2-hidden layer MLPs with 15 nodes per layer. In each case, a training set of 35,000 vectors and a separate test set of 35,000 vectors were used.*

simplicity we used the iid model, while some gain in performance may be achieved if the first-order Markov model is used [32]. In the case of Viterbi algorithm, a decoding delay of $10M_b = 20$ bits were used. The performance of the three types of decoders are compared in Figs 5.6, 5.7, and 5.8. In all the cases, the same encoder (and the same index assignment, optimized to VA hard-decoder at each CSNR) and channel have been used. Note that the recursive decoder in each case uses a delay equal to

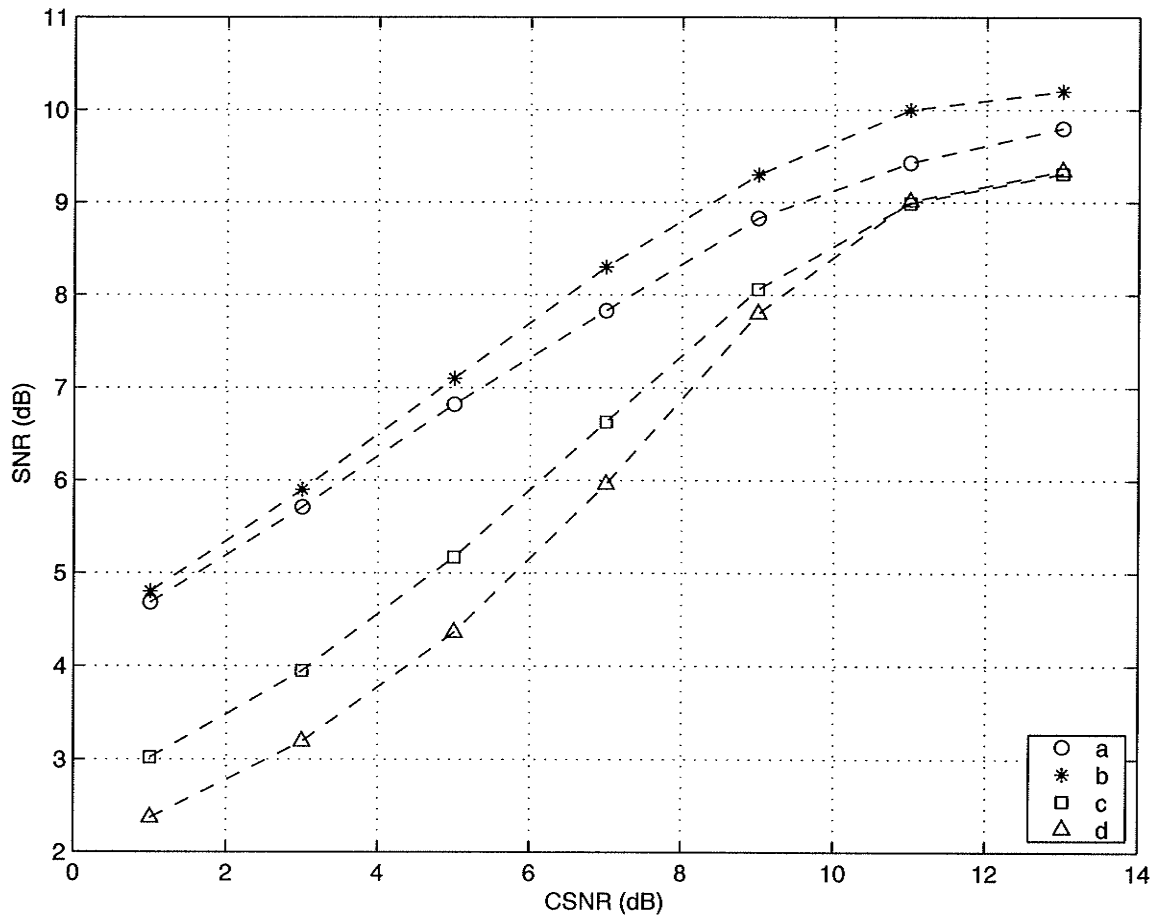


Figure 5.7: *Performance comparison of soft and hard decoding for d* = 3, *N* = 8*: (a) SB-SD(3,9) using MLP, (b) SB-SD(3,9) using the analytical equation (5.17), (c) R-SD(3), and (d) VA-HD(20). In each case a training set of 40,000 vectors and a separate test set of 40,000 vectors were used. MLP-based sliding-block decoder is a 2-hidden layer MLP with 20 nodes per layer.*

the channel memory. This corresponds to 2 bits in the case of d=2, N=4, 3 bits in the case of d=3, N=8 and 4 bits in the case of d=4, N=16. Skoglund [32] observed that a delay longer than channel memory resulted in almost no improvement in the performance of the recursive soft-decoder. In the results shown here, the sliding-block decoders also use the same delay, except in Fig. 5.6, where the results for delays of 10 bits (curve a) and 4 bits (curve d) are also shown. In all three figures, the sliding-block soft decoder appears to outperform the recursive soft-decoder, with the

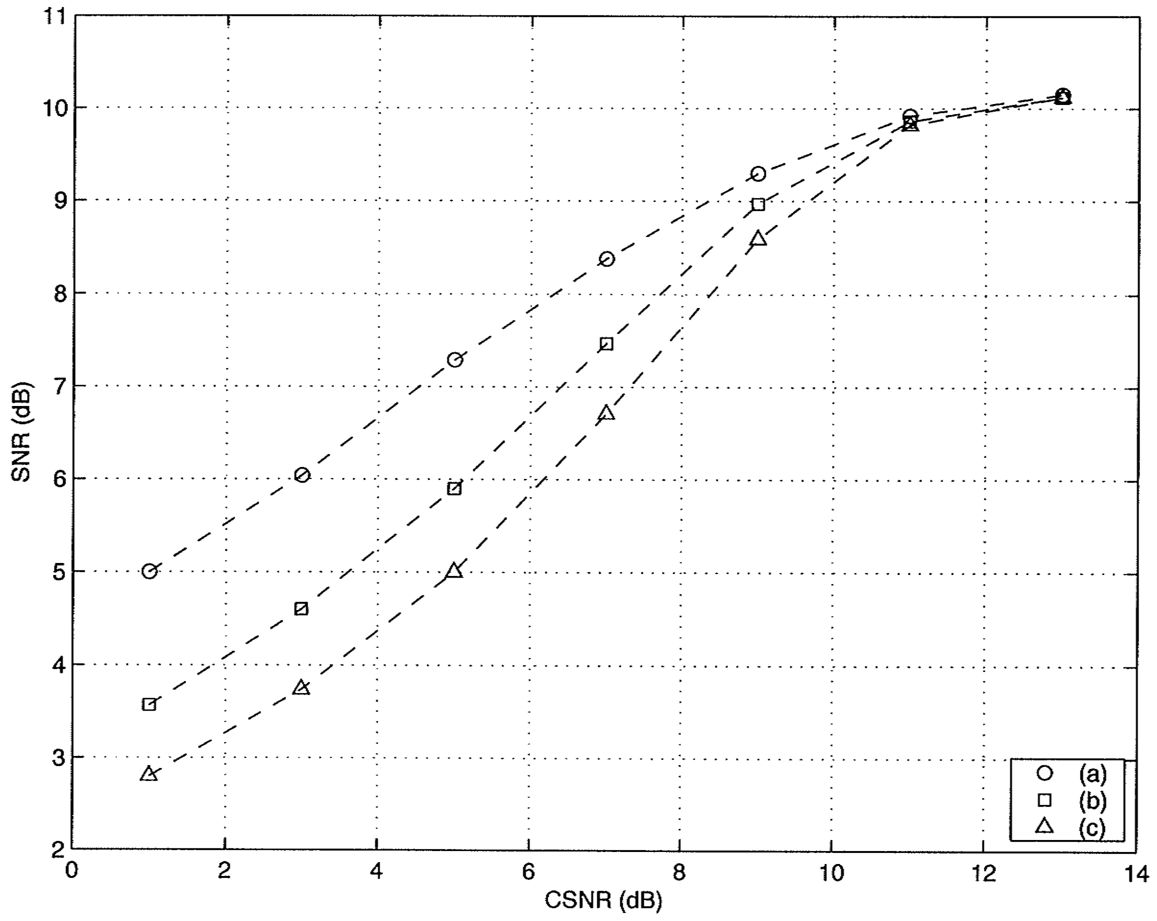


Figure 5.8: *Performance comparison of soft and hard decoding for* $d = 4$, $N = 16$: *(a) SB-SD(4,12) using MLP, (b) R-SD(4), and (c) VA-HD(20). In each case a training set of 50,000 vectors and a separate test set of 50,000 vectors were used. The sliding-block decoder is a 2-hidden layer MLP with 20 nodes per layer.*

largest improvement appearing in the case of two-dimensional VQ where the residual redundancy is high. Also shown in Fig. 5.6 (curve d) and Fig. 5.7 (curve b) is the performance of the optimal sliding-block decoder in each case, with block sizes of 6-bits (delay of 2 bits) and 9-bits (delay of 3-bits) respectively, evaluated using the exact expression in (5.17).

It striking that the relative performance of the sliding-block decoder (based on MLP approximation) increases at low CSNRs. We believe that this improvement is

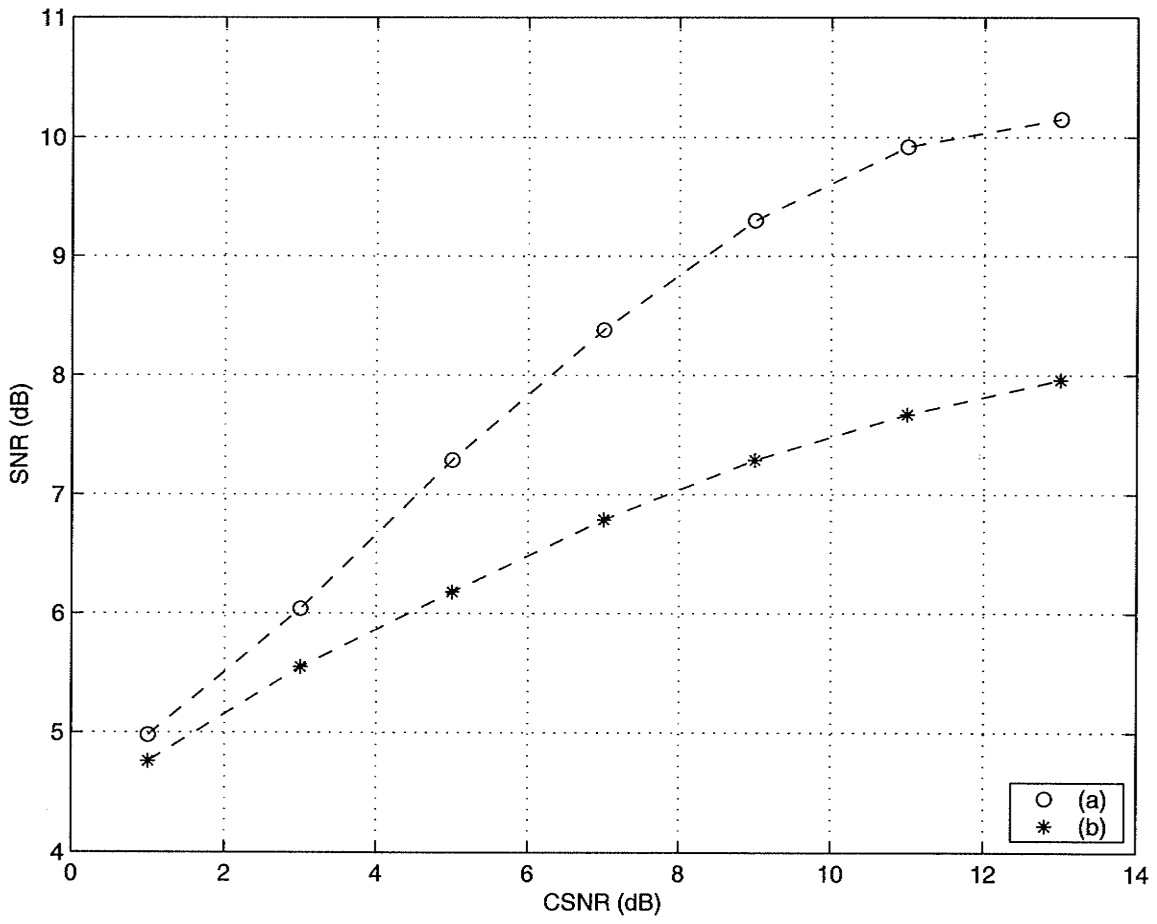


Figure 5.9: *Performance comparison of linear and non-linear sliding-block decoders for* $d = 4$, $N = 16$ *: (a) non-linear decoder (same as (a) in Fig. 5.8) with* $K_0 = 12$ *bits and (b) optimal linear decoder with* $K_0 = 12$ *bits . In each case a training set of 50,000 vectors and a separate test set of 50,000 vectors were used.*

due to the reduction of complexity of the optimal decoder mapping with the decreasing CSNR. In Section 5.5, we showed that as CSNR $\rightarrow$ 0, the optimal decoder tends to a linear mapping. This is also supported by the experimental results shown in Fig. 5.9, which indicate that the relative performance of the optimal linear decoder approaches that of the non-linear MLP decoder with identical block-size, as the CSNR is decreased. The conjecture is further strengthened by the results presented below.

It is possible to design COVQs for ISI channels by iteratively optimizing the encoder and decoder to each other as done in GLA. Fig. 5.10 shows the performance of COVQs based on different decoders, for the case of $d = 4$ and $N = 16$. Note that, as the recursive soft-decoder is time-variant, a fixed encoder is obtained in this case by using the time-averaged distortion measure [32]

$$\bar{D} = \lim_{n_{max} \to \infty} \frac{1}{n_{max}} \sum_{n=1}^{n_{max}} E\|\mathbf{X}_n - \hat{\mathbf{X}}_n(\mathbf{Y}_1, \ldots, \mathbf{Y}_{n+k})\|^2. \qquad (5.23)$$

This is not necessary in the case of the sliding-block decoder.

We next investigate the performance of sliding-block decoder when the channel input is uncorrelated. In order to do so, we consider VQ with $d = 4$ and $N = 16$ of an iid Gaussian process with variance 1.0. Note that, in this case, the residual redundancy is insignificant and the decoder memory solely serves to compensate for ISI. Fig. 5.11 shows the performance of various decoding strategies for iid source over the channel given by (5.22) (same encoder used in all cases). It is noticeable that the relative performance of the sliding block-decoder is poor at high CSNR. This error can be reduced by increasing the number of hidden nodes in the MLP. The improvement at low CSNR may be due to the fact that optimal decoder, which tends to a linear function as CSNR $\to 0$, is better approximated by the estimated MLP.

The complexity of the optimal sliding block decoder increases with N, making the estimation task difficult. This fact is apparent from the result shown in Table 5.4. In this case, the quantization of G-M source with $d = 3$ and $N = 128$ is considered, and as many as 60 nodes have been used in hidden layers. Still, the performance of the estimated decoder on a near noise free channel is much poorer compared to the performance of the Viterbi hard-decoder (nearly 5.5 dB below in SNR at 13 dB CSNR) . We have not attempted to increase the number of hidden nodes further, but it is clear that the number required for obtaining performance comparable with hard-

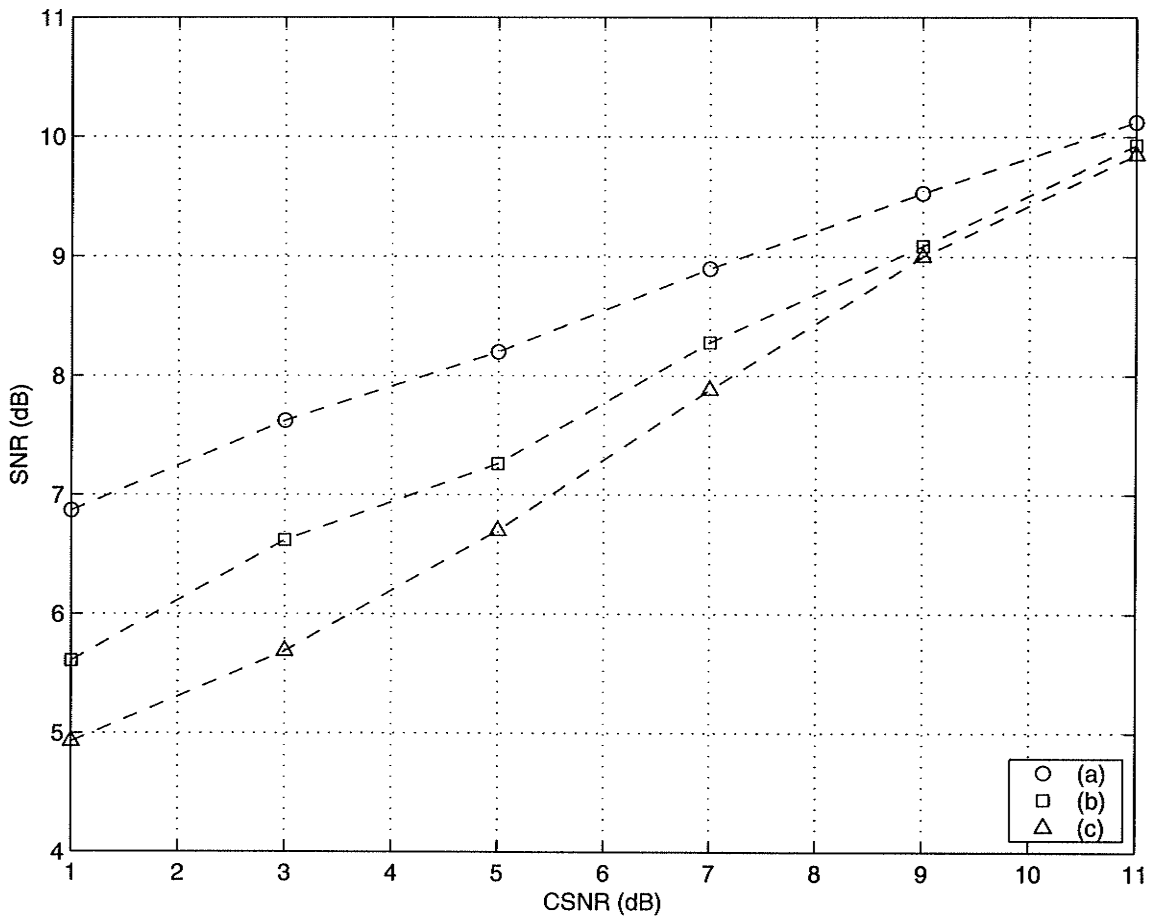


Figure 5.10: *Performance comparison of COVQ based on soft and hard decoding ($d = 4$, $N = 16$): (a) SB-SD(4,12) (c) R-SD(4), and (d) VA-HD(20). In each case a training set of 50,000 vectors was used for design and a separate set of 50,000 vectors were used for testing. Sliding-block decoder was a 2-hidden layer MLP with 20 hidden-nodes per layer.*

decoding at high CSNRs can be very high. Encouragingly, the relative performance of the estimated sliding block decoder improves significantly as the CSNR is lowered (over 2 dB better in SNR than hard-decoding at 1 dB CSNR).

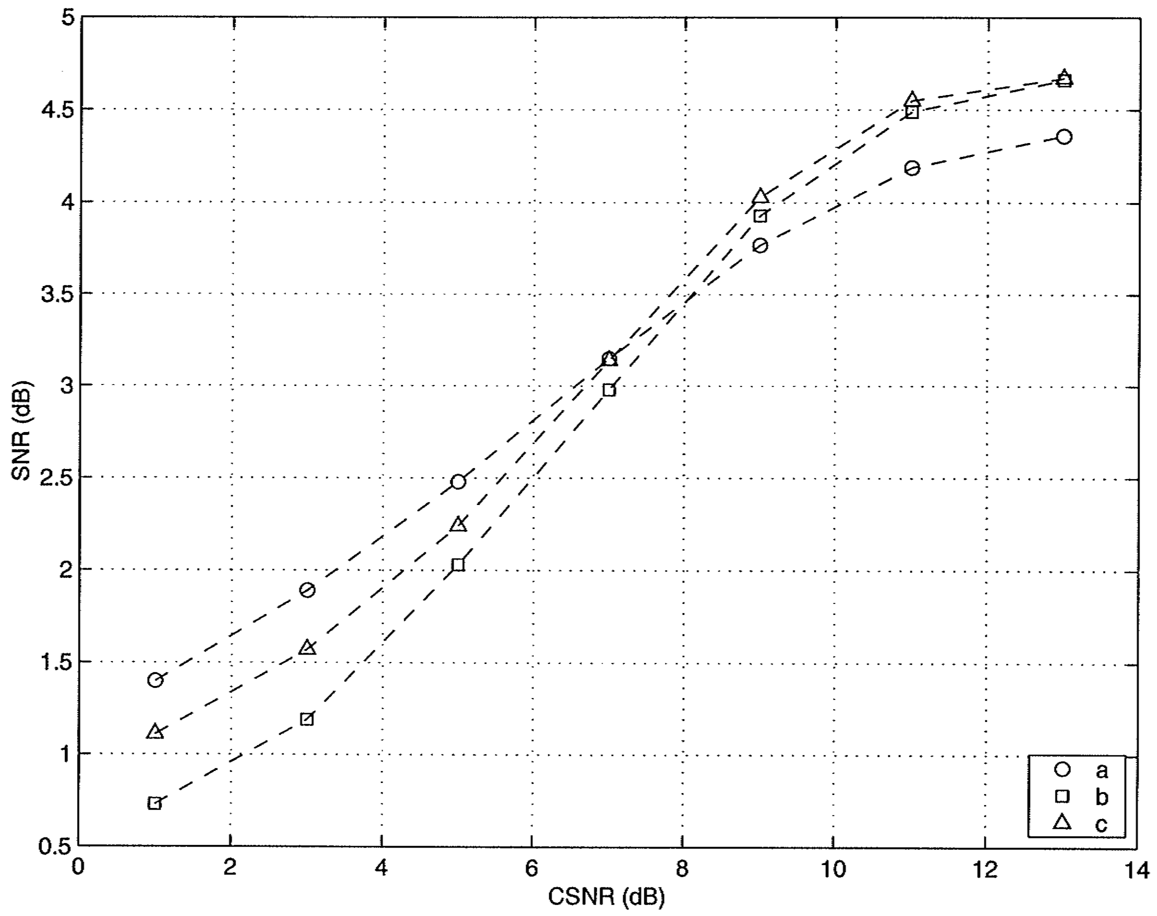


Figure 5.11: *Performance of various decoders ($d = 4, N = 16$) for iid source and linear Gaussian channel. (a) SB-SD(4,12), (b) R-SD(4), and (c) VA-HD(20). In each case a training set of 50,000 vectors was used for designing and a separate set of 50,000 vectors were used for testing. Sliding-block decoder was a 2-hidden layer MLP with 20 hidden-nodes per layer.*

$n_H(n_T)$	SNR 13 dB	1 dB
20 (50000)	7.67	
30 (50000)	9.36	
60 (80000)	11.30	6.59
VA-HD(20)	16.79	4.23

Table 5.4: *SNR performance of MLP sliding-block decoder SB-SD(7,21) and Viterbi hard-decoder, over channel given by (5.22) for G-M source, $d = 3$, $N = 128$. n_H - number of hidden nodes, n_T- training/test set size.*

5.8 Summary

In this chapter, we investigated soft-decoding based VQ for ISI channels. In particular, we studied a sliding-block decoder, which is a non-linear time-invariant filter estimated with the MMSE criterion. We used the MLP as the filter, which can be readily estimated via back-propagation algorithm. Simulation results were presented, which compared the sliding block decoder with both recursive soft decoding and hard decoding based on Viterbi channel equalization. In the cases we considered ($N \leq 16$), the estimated sliding-block decoder convincingly outperformed both the hard-decoder and the recursive soft decoder, particularly when the CSNR is low. On the negative side, the number of hidden nodes required in the MLP decoder for good approximation likely becomes unmanageably large, when the encoder resolution N is increased. In this context, we showed that the optimal sliding-block decoder for the Gaussian channel approaches a linear function as CSNR $\to 0$. This is encouraging as, even for large N, it is not hard to estimate a near optimal sliding-block decoder for highly noisy channels.

Chapter 6

Conclusions and Future Work

6.1 Summary and Conclusions

In this thesis, we have investigated the problem of designing channel optimized vector quantizers when memory is present in either the quantizer or the channel. Main contributions of this work can be summarized as follows.

- An algorithm for designing PVQ based on linear prediction for noisy channels was introduced. This algorithm iteratively improves an initial system to a given channel, by updating the predictor, the residual encoder, the local decoder, and the decoder at the receiver. Performance of the systems designed using this algorithm was investigated through simulation experiments, and both hard and soft decoding were considered. It was demonstrated that predictive VQ obtained by the new algorithm significantly outperforms memoryless VQ operating at the same rate (gains in SNR of about 2 dB with hard-decoding and about 3 dB with soft-decoding). We also found that the designs obtained by the new algorithm perform nearly identically to those obtained by gradient-search optimization algorithms in [22]. This observation is consistent with the previous observations made in the case of designing PVQs for noise free channels.

- The problem of designing FSVQ for noisy channels was investigated. The starting point of our work was the observation that a finite-state decoder is inappropriate for noisy channels, as it cannot track the encoder state-machine on the basis of noisy channel outputs. We demonstrated that the mean square error of an FSVQ increased catastrophically as the channel noise level was increased. To this end, we proposed a robust decoder for a given FSVQ encoder and derived a time-recursive algorithm for its implementation. The basic idea behind this decoder is to express the optimal reconstruction vector at a given time as a function of all the channel outputs observed up to that time. In effect, the decoder does not attempt to determine the state of the encoder, but considers that all states are possible with some probability. This decoder may be viewed as an infinite state decoder. Simulation results demonstrate that in contrast to a finite-state decoder, the new decoder exhibits graceful degradation of performance with increasing channel noise. We also incorporated this decoder in iteratively designed channel optimized FSVQ. These FSVQs always outperformed the memoryless COVQ. The gain in SNR was typically close 1 dB at very low CSNR, while much higher gains were obtained when the CSNR is high.

- Soft decoding in vector quantizers operating over channels with inter-symbol interference was studied. The problem was identified as one of estimating a random vector based on correlated observations (channel outputs). In this context, we studied a sliding-block soft-decoder, which is essentially a non-linear time-invariant filter estimated by MMSE criterion. As a practical implementation, we investigated the estimation of the decoder using multi-layer perceptrons

(MLP). Among other things, we demonstrated that sliding-block decoder significantly outperformed hard-decoding systems based on MLSE (Viterbi) channel equalization as the channel gets noisier. We also found the sliding-block decoder to outperform recursive soft decoding, previously investigated by Skoglund [32]. A potential problem with the sliding-block approach appeared to be the difficulty of estimating the decoder, when the encoder resolution N was high. With MLP implementations, the number of hidden nodes required for good approximation increases rapidly as N is increased. However, the problem seemed to be much less severe when the channel was very noisy. In this connection, we showed that, as the CSNR is reduced, the optimal sliding-block decoder for Gaussian channel approached a linear mapping. Consequently, the estimation of the decoder becomes easier for highly noisy channels.

6.2 Future Work

We mainly confined our attention to quantizing Gauss-Markov data since it is commonly used as a bench mark in judging the performance of coding algorithms. A natural extension of this work could be the application of the algorithms developed here to designing quantizers for real world signals. This would allow the evaluation of the performance of various designs based on subjective criteria. For example, both PVQ and FSVQ are strong candidates for low bit rate speech coding over wireless channels. In this context, coding of LSP parameters [54], [55] is an interesting application.

Another important avenue of exploration is to investigate the possibility of using function estimation techniques other than the MLP for sliding-block decoding. Simpler regression models, such as *additive models*, may reduce the computational burden associated with parameter estimation. In particular, *projection-pursuit networks* (PPR) have shown to yield sparser representations than MLPs in some cases. Another technique worth considering is *support vector machines.* We note here that a study in this direction would not only benefit soft VQ decoding, but may also benefit similar problems such as channel equalization.

In this dissertation, we have considered vector quantizers which act as joint-source channel coders. The benefit of these coders over comparable traditional tandem coding schemes (that is, schemes with separate source coding and channel coding) remains to be investigated. In certain situations, it has been found that COVQ is better only when the allowable delay is below a certain threshold [78].

Appendix A

MMSE Estimator

In this Appendix, the solution to the MMSE estimation problem is derived. Let $\mathbf{X} \in \mathbb{R}^d$ and $\mathbf{Y} \in \mathbb{R}^L$ be two random vectors with joint pdf $p(\mathbf{x}, \mathbf{y})$. Given the observed vector $\mathbf{Y}$, we wish to find an estimate $\hat{\mathbf{X}} = \delta(\mathbf{Y})$ for the unobserved vector $\mathbf{X}$, such that MSE $E\|\mathbf{X} - \delta(\mathbf{Y})\|^2$ is minimized. Consider

$$\begin{aligned} E\|\mathbf{X} - \delta(\mathbf{Y})\|^2 &= \int_{\mathbb{R}^d}\int_{\mathbb{R}^L} \|\mathbf{x} - \delta(\mathbf{y})\|^2 p(\mathbf{x}, \mathbf{y}) d\mathbf{x} d\mathbf{y} \\ &= \int_{\mathbb{R}^L}\int_{\mathbb{R}^d} \|\mathbf{x} - \delta(\mathbf{y})\|^2 p(\mathbf{x}|\mathbf{y}) d\mathbf{x} p(\mathbf{y}) d\mathbf{y} \\ &= \int_{\mathbb{R}^L} E\{\|\mathbf{X} - \delta(\mathbf{y})\|^2 | \mathbf{y}\} p(\mathbf{y}) d\mathbf{y} \end{aligned} \tag{A.1}$$

Since $E\{\|\mathbf{X} - \delta(\mathbf{y})\|^2|\mathbf{y}\}$ and $p(\mathbf{y})$ are non-negative, the MSE is minimized by minimizing the conditional expectation $\rho[\delta(\mathbf{y})] = E\{\|\mathbf{X} - \delta(\mathbf{y})\|^2|\mathbf{y}\}$ for every $\mathbf{y}$. That is

$$\delta^*(\mathbf{y}) = \arg\min_{\delta} \rho[\delta(\mathbf{y})]. \tag{A.2}$$

The desired solution is obtained by

$$\nabla_\delta \rho[\delta(\mathbf{y})] = \mathbf{0}, \tag{A.3}$$

which gives

$$\begin{aligned} -2\int_{\mathbb{R}^d} (\mathbf{x} - \delta^*(\mathbf{y}))p(\mathbf{x}|\mathbf{y})d\mathbf{x} = \mathbf{0} \\ \delta^*(\mathbf{y}) = E\{\mathbf{X}|\mathbf{Y} = \mathbf{y}\}. \end{aligned} \tag{A.4}$$

Also, $\nabla^2_\delta \rho[\delta(\mathbf{y})] = 2$ shows that the obtained solution is indeed a minimum.

The above result implies that the optimal estimator δ, minimizing $E\|A(\mathbf{X} - \delta(\mathbf{Y}))\|^2$, where A is a $d \times d$ constant matrix, is also given by (A.4), since

$$\begin{aligned} A\delta^*(\mathbf{y}) &= \arg\min_{A\delta} E\{\|A\mathbf{X} - A\delta(\mathbf{y})\|^2|\mathbf{y}\} \\ &= E\{A\mathbf{X}|\mathbf{Y} = \mathbf{y}\} \\ &= AE\{\mathbf{X}|\mathbf{Y} = \mathbf{y}\}. \end{aligned} \tag{A.5}$$

Appendix B

Derivation of (2.25)

When square error distortion measure is used, the average distortion of a noisy channel VQ can be readily expressed as the sum of quantization and channel error. To show this, we expand the average distortion in (2.14) as follows:

$$\begin{aligned} E\{D\} &= \sum_{i=1}^{N} \int_{\Omega_i} \|\mathbf{x}\|^2 p(\mathbf{x}) d\mathbf{x} \int_{\mathbb{R}^L} p(\mathbf{y}|\alpha_i) d\mathbf{y} \\ &\quad -2 \sum_{i=1}^{N} \int_{\Omega_i} \mathbf{x}^T p(\mathbf{x}) d\mathbf{x} \int_{\mathbb{R}^L} \delta_S(\mathbf{y}) p(\mathbf{y}|\alpha_i) d\mathbf{y} \\ &\quad + \sum_{i=1}^{N} \int_{\Omega_i} p(\mathbf{x}) d\mathbf{x} \int_{\mathbb{R}^L} \|\delta_S(\mathbf{y})\|^2 p(\mathbf{y}|\alpha_i) d\mathbf{y}. \end{aligned} \tag{B.1}$$

Now, defining the centroid of the i^{th} encoder cell $\int_{\Omega_i} \mathbf{x} p(\mathbf{x}) d\mathbf{x} / \int_{\Omega_i} p(\mathbf{x}) d\mathbf{x} = \mathbf{g}_i$ and $P(\alpha_i) = \Pr(\mathbf{x} \in \Omega_i)$, we get

$$\begin{aligned} E\{D\} &= \sum_{i=1}^{N} \int_{\Omega_i} \|\mathbf{x}\|^2 p(\mathbf{x}) d\mathbf{x} - 2 \sum_{i=1}^{N} P(\alpha_i) \mathbf{g}_i^T \int_{\mathbb{R}^L} \delta_S(\mathbf{y}) p(\mathbf{y}|\alpha_i) d\mathbf{y} \\ &\quad + \sum_{i=1}^{N} P(\alpha_i) \int_{\mathbb{R}^L} \|\delta_S(\mathbf{y})\|^2 p(\mathbf{y}|\alpha_i) d\mathbf{y}. \end{aligned} \tag{B.2}$$

Using the fact that

$$\int_{\Omega_i} \|\mathbf{x} - \mathbf{g}_i\|^2 p(\mathbf{x}) d\mathbf{x} = \int_{\Omega_i} (\|\mathbf{x}\|^2 - \|\mathbf{g}_i\|^2) p(\mathbf{x}) d\mathbf{x} \tag{B.3}$$

and after some manipulations, we end up with the desired result

$$E\{D\} = \sum_{i=1}^{N} \int_{\Omega_i} \|\mathbf{x} - \mathbf{g}_i\|^2 p(\mathbf{x}) d\mathbf{x} + \sum_{i=1}^{N} \int_{\mathbb{R}^L} \|\mathbf{x} - \delta_S(\mathbf{y})\|^2 p(\mathbf{y}|\alpha_i) P(\alpha_i) d\mathbf{y}. \tag{B.4}$$

Appendix C

Gauss-Markov Source

In this appendix, the first-order Gauss-Markov (G-M) source used for simulation studies in this dissertation is described. G-M source is widely used as a benchmark for comparing different source coding techniques and is a good model for real world data in many cases. For example, the correlation coefficient of 0.9 is typical of speech and image data.

A first-order G-M process is described by the auto-regressive relation

$$X_n = \rho X_{n-1} + U_n, \tag{C.1}$$

where $\rho = EX_n X_{n-1}/EX_n^2$ is the correlation coefficient and U_n is a stationary iid Gaussian process with mean zero and variance σ_U^2. Hence $EX_n = 0$ and $EX_n^2 = \sigma_U^2/(1-\rho^2)$ $(= \sigma_X^2)$. In our experiments, we have used $\sigma_U^2 = 1.0$ and hence $\sigma_X^2 = 5.26$.

The best achievable performance by any quantizer on this source, often referred to as *optimal performance theoretically attainable* (OPTA), can be found by computing its rate-distortion function [53]

$$D(R) = (1-\rho^2)2^{-2R}\sigma_X^2 \quad \text{for } R \geq \log_2(1+\rho). \tag{C.2}$$

For $R < \log_2(1+\rho)$, this gives a lower bound for $R(D)$.[1] When $\rho = 0.9$, (C.2) can be used to obtain $D(R)$ for $R > 0.9260$.

In order to compute the OPTA of the source over a noisy channel, we can use the Shannon's joint source-channel coding theorem [6]. That is, given a channel with *channel capacity* C, one may transmit up to RC bits by R uses of the channel. Hence, it follows that OPTA of the source over such a channel is $D(RC)$. Table C.1 shows OPTA (expressed in terms of SNR) for G-S source with $\rho = 0.9$ at $R = 1$ bit per sample, over a memoryless Gaussian channel with binary antipodal signaling.

CSNR (dB)	BER	OPTA (dB)
∞	0.0000	13.2331
12	3.43×10^{-5}	13.2297
10	0.0008	13.1766
8	0.0060	12.9145
6	0.0230	12.2820
4	0.0560	11.3585
2	0.1040	10.3339

Table C.1: *OPTA of Gauss-Markov source with* $\rho = 0.9$ *and* $\sigma_X^2 = 5.26$ *at* $R = 1$ *bit per sample over the memoryless Gaussian channel.*

[1] Actual values have to be obtained by parametric expressions [53].

Appendix D

Index Assignment for FSVQ

In this appendix, a distortion measure is derived, which can be used to optimize the index assignment (IA) of an FSVQ system to a given channel. The system considered here is an ordinary FSVQ and we use the notation defined in Fig. 4.3. Let the encoder partition of state s be given by $\Omega_s(i)$, where $i = 1, 2, \ldots, N$ and $s = 1, 2, \ldots, K$, such that $\cup_i^N \Omega_s(i) = \mathbb{R}^d$ and $\cap_i^N \Omega_s(i) = \emptyset$. Then, the MSE given by (4.13) may be written as

$$\begin{aligned} E\|\mathbf{X}_n - \hat{\mathbf{X}}_n\|^2 &= \sum_{s_n, i_n} \sum_{\hat{s}_n, j_n} P(s_n) \int_{\Omega_{s_n}(i_n)} \|\mathbf{x}_n - \hat{\mathbf{x}}_n\|^2 P(\hat{s}_n, j_n | s_n, i_n) p(\mathbf{x}_n | s_n) d\mathbf{x}_n \\ &= \sum_{s_n, i_n} \sum_{\hat{s}_n, j_n} P(s_n) P(\hat{s}_n | s_n) P(j_n | i_n) \int_{\Omega_{s_n}(i_n)} \|\mathbf{x}_n - \hat{\mathbf{x}}_n\|^2 p(\mathbf{x}_n | s_n) d\mathbf{x}_n. \end{aligned} \tag{D.1}$$

After expanding the square term inside the integral, the above expression becomes

$$\begin{aligned} E\|\mathbf{X}_n - \hat{\mathbf{X}}_n\|^2 &= \sum_{s_n, i_n} \int_{\Omega_{s_n}(i_n)} \|\mathbf{x}_n\|^2 p(\mathbf{x}_n|s_n) d\mathbf{x}_n \\ &- 2 \sum_{s_n, i_n} \sum_{\hat{s}_n, j_n} P(s_n) P(\hat{s}_n|s_n) P(j_n|i_n) \hat{\mathbf{x}}_n^T \int_{\Omega_{s_n}(i_n)} \mathbf{x}_n p(\mathbf{x}_n|s_n) d\mathbf{x}_n \\ &+ \sum_{s_n, i_n} \sum_{\hat{s}_n, j_n} P(s_n) P(\hat{s}_n|s_n) P(j_n|i_n) \|\hat{\mathbf{x}}_n\|^2 \int_{\Omega_{s_n}(i_n)} p(\mathbf{x}_n|s_n) d\mathbf{x}_n. \end{aligned} \tag{D.2}$$

Now, using the fact that

$$P_{s_n}(i_n) = \Pr(\mathbf{x}_n \in \Omega_{s_n}(i_n)|s_n) = \int_{\Omega_{s_n}(i_n)} p(\mathbf{x}_n|s_n) d\mathbf{x}_n \tag{D.3}$$

and

$$\mathbf{g}_{s_n}(i_n) = E\{\mathbf{x}_n|i_n, s_n\} = \frac{\int_{\Omega_{s_n}(i_n)} \mathbf{x}_n p(\mathbf{x}_n|s_n) d\mathbf{x}_n}{P_{s_n}(i_n)}, \tag{D.4}$$

and after rearranging the terms within the expression, it is straightforward to show that

$$E\|\mathbf{X}_n - \hat{\mathbf{X}}_n\|^2 = D_Q + D_C, \tag{D.5}$$

where

$$D_Q = \sum_{s_n, i_n} P(s_n) \int_{\Omega_{s_n}(i_n)} \|\mathbf{x}_n - \mathbf{g}_{s_n}(i_n)\|^2 p(\mathbf{x}_n|s_n) d\mathbf{x}_n, \tag{D.6}$$

$$D_C = \sum_{s_n, i_n} \sum_{\hat{s}_n, j_n} P(\hat{s}_n|s_n) P(j_n|i_n) P(s_n) \|\mathbf{g}_{s_n}(i_n) - \hat{\mathbf{x}}_n(\hat{s}_n, j_n)\|^2 P_{s_n}(i_n). \tag{D.7}$$

All but the conditional probabilities $P(\hat{s}_n|s_n)$ in the above expression solely depend on the encoder and hence can be computed (using training data) if the encoder is given. We next show how steady-state values of $P(\hat{s}_n|s_n)$ can be computed. To this end we write

$$P(\hat{s}_n|s_n) = \sum_{\hat{s}_{n-1}, j_{n-1}} P(\hat{s}_n|\hat{s}_{n-1}, j_{n-1}, s_{n-1}, i_{n-1}) P(\hat{s}_{n-1}, j_{n-1}|s_n). \tag{D.8}$$

Define the set of all (s_{n-1}, i_{n-1}) pairs which lead the state $s_n = s$ as

$$\mu(s) = \{(s_{n-1}, i_{n-1}) : f(s_{n-1}, i_{n-1}) = s\} \tag{D.9}$$

Since the decoder uses the same next-state rule as the encoder, it is also implied that $\mu(\hat{s}) = \{(\hat{s}_{n-1}, j_{n-1}) : f(\hat{s}_{n-1}, j_{n-1}) = \hat{s}\}$. Using this notation, we can write the above equation as

$$\begin{aligned} P(\hat{s}_n|s_n) &= \sum_{\mu(\hat{s}_n)} P(\hat{s}_{n-1}, j_{n-1}|s_n) \\ &= \sum_{\mu(\hat{s}_n)} \sum_{s_{n-1}, i_{n-1}} P(\hat{s}_{n-1}, j_{n-1}|s_{n-1}, i_{n-1}, s_n) P(s_{n-1}, i_{n-1}|s_n) \\ &= \sum_{\mu(\hat{s}_n)} \sum_{\mu(s_n)} P(\hat{s}_{n-1}|s_{n-1}) P(j_{n-1}|i_{n-1}) P(i_{n-1}|s_{n-1}) P(s_{n-1})/P(s_n), \end{aligned} \tag{D.10}$$

where we use the fact that the channel is memoryless. Since at steady-state $P(\hat{s}_{n-1}|s_{n-1})$ $= P(\hat{s}_n|s_n)$ the above expression gives us a redundant system of K^2 linear equations with K^2 unknowns $P(\hat{s}_n = s_i|s_n = s_j)$, $i, j = 1, 2, \ldots, K$. However this system can be easily solved considering $K - 1$ of these equations together with the equation

$$\sum_{s_n} \sum_{\hat{s}_n} P(\hat{s}_n|s_n) P(s_n) = 1. \tag{D.11}$$

Appendix E

OLT-FSVQ Design Algorithm

This appendix describes the omniscient labeled-transition FSVQ (OLT-FSVQ) design algorithm [49] used to obtain the next-state rule and the initial state encoders for designs described in Section 4.10. Let K be the number of states and N be the number of code vectors (encoders cells) per state.

Given: A training set $\mathcal{T} = \{\mathbf{x}_i\}_{i=1}^{n_T}$ of n_T vectors from the source.

Step 1: Design a memoryless VQ with a size K codebook (*e.g.*, use GLA [12]), using $\mathcal{T}$. This VQ defines the initial super codebook $\mathbb{C}^{(sup)} = \{\mathbf{c}_i^{(sup)}\}_{i=1}^{K}$. At this point *next-state rule* is

$$f^{(0)}(\mathbf{x}_n) = \mathbf{s}_{n+1} = \arg\min_{s} \|\mathbf{x}_n - \mathbf{c}_s^{(sup)}\|^2$$

Step 2: Classify each vector $\mathbf{x}_i$ in $\mathcal{T}$ into K sub-groups (sub-sequences) according to the state transition caused by of $\mathbf{x}_{i-1}$. That is

$$\mathcal{T}_s = \{\mathbf{x}_i : s = f(\mathbf{x}_{i-1})\text{and } \mathbf{x}_{i-1}, \mathbf{x}_i \in \mathcal{T}\}.$$

For each $s = 1, 2, \ldots, K$ use GLA to design a codebook $\mathbb{C}_s = \{\mathbf{c}_i^{(s)}\}_{i=1}^{N}$ with training set $\mathcal{T}_s$. These K codebooks are taken as the *state codebooks.*

Step 3: Next-state rule $f^{(0)}(\mathbf{x}_n)$ cannot be used by the decoder since $\mathbf{x}_n$ is not available at the decoder. Hence, replace $\mathbf{x}_n$ with $\hat{\mathbf{x}}_n = \mathbf{c}_{i_n}^{(s_n)}$, that is, modify the next state-rule as

$$f(s_n, i_n) = \mathbf{s}_{n+1} = \arg\min_{s} \|\mathbf{c}_{i_n}^{(s_n)} - \mathbf{c}_s^{(sup)}\|^2.$$

Appendix F

Derivation of (5.15)

We consider the finite memory channel characterized by (5.1), when the additive channel noise is iid Gaussian. The derivation given below is an extension of a similar result derived in [70] for the memoryless AWGN channel. Referring to (5.1), let

$$\mathbf{r}_n = f(\mathbf{s}_n, \mathbf{s}_{n-1}, \ldots, \mathbf{s}_{n-M}), \tag{F.1}$$

where $\mathbf{r}_n \in \mathbb{R}^L$. Next, consider the sliding-block decoder described by (5.13). If the channel noise $\mathbf{w}_n$ is independent of the channel input and is iid Gaussian,

$$\begin{aligned} p(\mathbf{u}_n|\mathbf{v}_n^{(i)}) &= \prod_{j=0}^{K_1+K_2} p_{\mathbf{W}}(\mathbf{y}_{n+K_2-j} - \mathbf{r}^{(i)}_{n+K_2-j}) \\ &= \frac{1}{(2\pi\sigma_W)^{L/2}} \exp\Big[-\frac{1}{2\sigma_W^2}\sum_{j=0}^{K_1+K_2} \|\mathbf{y}_{n+K_2-j} - \mathbf{r}^{(i)}_{n+K_2-j}\|^2\Big]. \end{aligned} \tag{F.2}$$

For notational simplicity, define

$$\bar{\mathbf{r}}_n = (\mathbf{r}^T_{n-K_1}, \ldots, \mathbf{r}^T_{n+K_2})^T, \tag{F.3}$$

where $\bar{\mathbf{r}}_n \in \mathbb{R}^{L(K_1+K_2+1)}$ and note that

$$\sum_{j=0}^{K_1+K_2} \|\mathbf{y}_{n+K_2-j} - \mathbf{r}^{(i)}_{n+K_2-j}\|^2 = \|\mathbf{u}_n - \bar{\mathbf{r}}^{(i)}_n\|^2. \tag{F.4}$$

Hence, (F.2) can be written as

$$\begin{aligned} p(\mathbf{u}_n|\mathbf{v}^{(i)}_n) &= \frac{1}{(2\pi\sigma_W)^{L/2}} \exp\left[-\frac{1}{2\sigma^2_W}\|\mathbf{u}_n - \bar{\mathbf{r}}^{(i)}_n\|^2\right] \\ &= p'_W(\mathbf{u}_n - \bar{\mathbf{r}}^{(i)}_n), \end{aligned} \tag{F.5}$$

where p'_W is a $L(K_1+K_2+1)$ dimensional density function with the covariance matrix $\operatorname{diag}(\sigma^2_W, \ldots, \sigma^2_W)$. Using the first-order Taylor series expansion, (F.5) may be expressed as

$$p'_W(\mathbf{u}_n - \bar{\mathbf{r}}^{(i)}_n) = p'_W(\mathbf{u}_n) - \nabla_{\mathbf{u}_n} p'_W(\mathbf{u}_n)\bar{\mathbf{r}}^{(i)}_n + o(\bar{\mathbf{r}}^{(i)}_n, \mathbf{u}_n), \tag{F.6}$$

where $o(\bar{\mathbf{r}}^{(i)}_n, \mathbf{u}_n)/\|\bar{\mathbf{r}}^{(i)}_n\| \to 0$ as $\|\bar{\mathbf{r}}^{(i)}_n\| \to 0$ and $\nabla_{\mathbf{u}_n} p'_W(\mathbf{u}_n) = (-1/\sigma^2_W)p'_W(\mathbf{u}_n)\mathbf{u}^T_n$. By substituting these results in (5.13) and assuming (without loss of generality) that $E\{\bar{\mathbf{r}}^{(i)}_n\} = \mathbf{0}$, we obtain

$$\phi(\mathbf{u}_n) = \frac{\sum_i P(\mathbf{v}^{(i)}_n)\mathbf{g}'_i\bar{\mathbf{r}}^{(i)\,T}_n \mathbf{u}_n/\sigma^2_W + \sum_i P(\mathbf{v}^{(i)}_n)\mathbf{g}'_i o(\bar{\mathbf{r}}^{(i)}_n, \mathbf{u}_n)/p'_W(\mathbf{u}_n)}{1 + \sum_i P(\mathbf{v}^{(i)}_n) o(\bar{\mathbf{r}}^{(i)}_n, \mathbf{u}_n)/p'_W(\mathbf{u}_n)}. \tag{F.7}$$

Let the average signal power at the output of the channel $E\{\mathbf{r}^T_n\mathbf{r}_n\} = E_0$. Then it follows that when the channel output is stationary, $E\{\bar{\mathbf{r}}_n\bar{\mathbf{r}}^T_n\} = K_0E_0$, where $K_0 = L(K_1+K_2+1)$. Also let

$$G = \sum_i P(\mathbf{v}^{(i)}_n)\mathbf{g}'_i\bar{\mathbf{r}}^{(i)\,T}_n/\sigma^2_W, \tag{F.8}$$

which is a $d \times K_0 L$ matrix that depends on the source, encoder, and channel. Our goal is to establish that

$$\lim_{E_0 \to 0} \frac{\|\phi(\mathbf{u}_n) - G\mathbf{u}_n\|}{\sqrt{E_0}} = 0. \tag{F.9}$$

In order to do this, consider

$$\begin{aligned} \frac{\|\phi(\mathbf{u}_n) - G\mathbf{u}_n\|}{\sqrt{K_0 E_0}} &= \frac{\|\sum_i P(\mathbf{v}_n^{(i)})\mathbf{g}_i' o(\bar{\mathbf{r}}_n^{(i)}, \mathbf{u}_n) - G\mathbf{u}_n\Big(\sum_i P(\mathbf{v}_n^{(i)}) o(\bar{\mathbf{r}}_n^{(i)}, \mathbf{u}_n)\Big)\|}{p_W'(\mathbf{u}_n)\sqrt{K_0 E_0}|1 + \sum_i P(\mathbf{v}_n^{(i)}) o(\bar{\mathbf{r}}_n^{(i)}, \mathbf{u}_n)/p_W'(\mathbf{u}_n)|} \\ &\le \frac{\sum_i P(\mathbf{v}_n^{(i)})\|\mathbf{g}_i'\||o(\bar{\mathbf{r}}_n^{(i)}, \mathbf{u}_n)| - \|G\mathbf{u}_n\|\Big(\sum_i P(\mathbf{v}_n^{(i)})|o(\bar{\mathbf{r}}_n^{(i)}, \mathbf{u}_n)|\Big)}{p_W'(\mathbf{u}_n)\sqrt{K_0 E_0}|1 + \sum_i P(\mathbf{v}_n^{(i)}) o(\bar{\mathbf{r}}_n^{(i)}, \mathbf{u}_n)/p_W'(\mathbf{u}_n)|}. \end{aligned} \tag{F.10}$$

The desired result is obtained by showing that

$$\lim_{E_0 \to 0} \frac{|o(\bar{\mathbf{r}}_n^{(i)}, \mathbf{u}_n)|}{\sqrt{K_0 E_0}} = 0, \tag{F.11}$$

which follows from

$$\begin{aligned} \frac{|o(\bar{\mathbf{r}}_n^{(i)}, \mathbf{u}_n)|}{\sqrt{K_0 E_0}} &= \frac{|o(\bar{\mathbf{r}}_n^{(i)}, \mathbf{u}_n)|}{\|\bar{\mathbf{r}}_n^{(i)}\|} \frac{\|\bar{\mathbf{r}}_n^{(i)}\|}{\sqrt{K_0 E_0}} \\ &\le \frac{|o(\bar{\mathbf{r}}_n^{(i)}, \mathbf{u}_n)|}{\|\bar{\mathbf{r}}_n^{(i)}\|} \frac{1}{\sqrt{P(\mathbf{v}_n^{(i)})}}. \end{aligned} \tag{F.12}$$

Here, we use the fact that

$$K_0 E_0 = \sum_i P(\mathbf{v}_n^{(i)})\|\bar{\mathbf{r}}_n^{(i)}\|^2, \tag{F.13}$$

$$\sqrt{K_0 E_0} \ge \sqrt{P(\mathbf{v}_n^{(i)})}\|\bar{\mathbf{r}}_n^{(i)}\|. \tag{F.14}$$

Bibliography

[1] R. M. Gray, "Vector quantization," *IEEE ASSP Magazine*, pp. 4–29, April 1984.

[2] R. M. Gray and D. L. Neuhoff, "Quantization," *IEEE Transactions on Information Theory*, vol. 44, pp. 50–55, October 1998.

[3] A. Gersho and R. M. Gray, *Vector Quantization.* Kluwer Academic Publishers, 1992.

[4] A. Buzo, J. A. H. Gray, R. M. Gray, and J. D. Markel, "Speech coding based upon vector quantization: A review," *IEEE Transactions on Acoustic, Speech, and Signal Processing*, vol. ASSP-28, no. 5, pp. 562–574, 1980.

[5] N. M. Nasrabadi and R. A. King, "Image coding using vector quantization: A review," *IEEE Transactions on Communications*, vol. 36, pp. 957–971, August 1988.

[6] C. E. Shannon, "A mathematical theory of communication," *Bell Systems Journal*, vol. 27, pp. 379–423, 623–656, 1948.

[7] R. E. V. Dyck and D. J. Miller, "Transport of wireless video using separate, concatenated and joint source-channel coding," *Proceedings of the IEEE*, vol. 87, pp. 1734–1750, October 1999.

[8] C. E. Shannon, "Coding theorems for a discrete source with a fidelity criterion," *IRE National Convention Record, Part 4*, pp. 142–163, 1959.

[9] T. Berger and J. D. Gibson, "Lossy source coding," *IEEE Transactions on Information Theory*, vol. 44, pp. 2693–2733, October 1998.

[10] S. P. Lloyd, "Least square quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, pp. 129–137, March 1982.

[11] J. Max, "Quantizing for minimum distortion," *IRE Transaction on Information Theory*, vol. IT-6, pp. 7–12, March 1960.

[12] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communications*, vol. COM-28, pp. 84–95, January 1980.

[13] A. J. Kurtenbach and P. A. Wintz, "Quantizing for noisy channels," *IEEE Transactions on Communications*, vol. COM-17, pp. 291–302, April 1969.

[14] N. Farvardin and V. Vaishampayan, "Optimal quantizer design for noisy channels: an approach to combined source-channel coding," *IEEE Transactions on Information Theory*, vol. 33, pp. 827–838, November 1987.

[15] E. Ayanoglu and R. M. Gray, "The design of joint source and channel trellis waveform coders," *IEEE Transactions on Information Theory*, vol. IT-33, pp. 855–865, November 1987.

[16] H. Kumazawa, M. Kasahara, and T. Namekawa, "A construction of vector quantizers for noisy channels," *Electronics and Communications in Japan*, vol. 67, pp. 1–8, January 1984.

[17] N. Farvardin, "On the performance and complexity of channel-optimized vector quantizers," *IEEE Transactions on Information Theory*, vol. 37, pp. 155–160, January 1991.

[18] J. Rosebrock and P. W. Besslich, "Shape-gain vector quantization for noisy channels with applications to image coding," *IEEE Transactions on Selected Areas In Communications*, vol. 10, pp. 918–925, June 1992.

[19] N. Phamdo, N. Farvardin, and T. Moriya, "A unified approach to tree-structured and multi-stage vector quantization for noisy channels," *IEEE Transactions on Information Theory*, vol. 39, pp. 835–850, May 1993.

[20] Y. Hussain, *Design and Performance Evaluation of a Class of Finite-state Vector Quantizers.* Ph.D. Thesis, University of Maryland, College Park, 1992.

[21] Y. Hussain and N. Farvardin, "Finite-state vector quantization over noisy channels and its application to LSP parameters," *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. 133–136, 1992.

[22] J. Lindén, "Channel optimized predictive VQ," *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. 681–684, March 1999.

[23] J. Lindén, *Interframe Quantization for Noisy Channels.* Ph.D. Thesis, Chalmers University of Technology, Goteborg, Sweden, 1998.

[24] V. A. Vaishampayan and N. Farvardin, "Joint design of block source codes and modulation signal sets," *IEEE Transactions on Information Theory*, vol. 38, pp. 1230–1248, July 1992.

[25] F. Liu, P. Ho, and V. Cuperman, "Joint source and channel coding using a non-linear receiver," in *Proc. IEEE International Conference on Communications (ICC'93)*, pp. 1502–1507, 1993.

[26] F. Liu, P. Ho, and V. Cuperman, "Sequential reconstruction of vector quantized signals transmitted over Rayleigh fading channel," in *Proc. IEEE International Conference on Communications (ICC'94)*, pp. 23–27, 1994.

[27] M. Skoglund and P. Hedelin, "Hadamard-based soft-decoding for vector quantization over noisy channels," *IEEE Transactions on Information Theory*, vol. IT-28, pp. 167–186, March 1999.

[28] M. Skoglund and T. Ottoson, "Soft multi-user decoding for vector quantization over a CDMA channel," *IEEE Transactions on Communications*, vol. 46, pp. 327–337, March 1998.

[29] N. Phamdo and F. Alajaji, "Soft-decision demodulation design for COVQ over white, colored, and ISI Gaussian channels," *IEEE Transactions on Communications*, vol. 48, pp. 1499–1506, September 2000.

[30] F. Alajaji and N. Phamdo, "Soft-decision COVQ Rayleigh fading channels," *IEEE Communication Letters*, vol. 2, pp. 162–164, June 1998.

[31] V. Kafedziski and D. Morell, "Vector quantization over Gaussian channels with memory," in *Proc. IEEE International Conference on Communications (ICC'95)*, pp. 1433–1437, 1995.

[32] M. Skoglund, "Soft decoding for vector quantization over noisy channels with memory," *IEEE Transactions on Information Theory*, vol. 45, pp. 1293–1307, May 1999.

[33] M. Skoglund, "Bit-estimate based decoding for vector quantization over noisy channels with intersymbol interference," *IEEE Transactions on Communications*, vol. 48, pp. 1309–1317, August 2000.

[34] N. Phamdo and F. Alajaji, "Soft-decision demodulation design for covq over white, colored and isi gaussian channels," *IEEE Transactions on Communications*, vol. 48, pp. 1499–1506, September 2000.

[35] H. S. Wang, *Finite-state Modeling, Capacity, and Joint Source/Channel Coding for Time-Varying Channels.* Ph.D. Thesis, Rutgers, The State University of New Jersey- New Brunswick, 1992.

[36] T. M. Duman and M. Salehi, "Optimal quantization for finite-state channels," *IEEE Information theory*, vol. 43, pp. 758–765, March 1997.

[37] H. Jafarkhani and N. Farvardin, "Design of channel optimized vector quantizers in the presence of channel mismatch," *IEEE Transactions on Communications*, pp. 118–124, January 2000.

[38] J. D. Marca and N. Jayant, "An algorithm for assigning binary indices to the codevectors of a multi-dimensional quantizer," in *Proc. IEEE International Conference on Communications (ICC'87)*, pp. 1128–1132, 1987.

[39] K. Zeger and A. Gersho, "Pseudo-gray coding," *IEEE Transactions on Communications*, vol. 38, pp. 2147–2158, December 1990.

[40] N. Farvardin, "A study of vector quantization for noisy channels," *IEEE Transactions on Information Theory*, vol. 36, pp. 799–809, July 1990.

[41] K. Zeger and V. Manzella, "Asymptotic bounds on optimal noisy channel quantization via random coding," *IEEE Transactions on Information Theory*, vol. 40, pp. 1926–1938, November 1994.

[42] T. Linder, G. Lugosi, and K. Zeger, "Empirical quantizer design in the presence of source noise or channel noise," *IEEE Transactions on Information Theory*, vol. 43, pp. 612–623, March 1997.

[43] D. Miller and K. Rose, "Combined source-channel vector quantization using deterministic annealing," *IEEE Transactions on Communications*, vol. 42, pp. 347–356, February 1994.

[44] T. Eriksson, J. Lindén, and J. Skoglund, "Interframe LSF quantization for noisy channels," *IEEE Transactions on Speech and Audio Processing*, vol. 7, pp. 495–509, September 1999.

[45] J. Skoglund and J. Lindén, "Predictive VQ for noisy channel spectrum coding: AR or MA ?," *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. 1351–1354, April 1997.

[46] V. Cupreman and A. Gersho, "Vector predictive coding of speech at 16 kbits/s," *IEEE Transactions on Communications*, vol. COM-33, no. 7, pp. 83–89, 1985.

[47] K.-Y. Chang and R. W. Donaldson, "Analysis, optimization, and sensitivity study of differential PCM systems operating on noisy communication channels," *IEEE Transactions on Communications*, vol. COM-20, pp. 338–350, June 1972.

[48] N. S. Jayant and P. Noll, *Digital Coding of Waveforms: Principles and Applications to Speech and Video.* Prentice-Hall, 1984.

[49] J. Foster, R. M. Gray, and M. O. Dunham, "Finite-state vector quantization for waveform coding," *IEEE Transactions on Information Theory*, vol. IT-31, pp. 348–359, May 1985.

[50] K. Abend and B. D. Fritchman, "Statistical detection for communication channels with intersymbol interference," *Proceedings of the IEEE*, vol. 58, pp. 779–785, May 1970.

[51] B. D. O. Anderson and J. B. Moore, *Optimal Filtering.* Prentice Hall, 1979.

[52] K. Sayood and J. C. Brokenhagen, "Use of residual redundancy in the design of joint source/channel coders," *IEEE Transactions on Communications*, vol. 39, pp. 838–846, June 1991.

[53] T. Berger, *Rate Distortion Theory: A Mathematical Basis for Data Compression.* Prentice-Hall, 1971.

[54] F. K. Soong and B.-H. Juang, "Line spectrum pair (LSP) and speech data compression," *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 1.10.1–1.10.4, 1984.

[55] K. K. Paliwal and B. S. Atal, "Efficient vector quantization of LPC parameters at 24 bits/frame," *IEEE Transactions on Speech and Audio Processing*, vol. 1, pp. 3–14, January 1993.

[56] H.-M. Hang and J. W. Woods, "Predictive quantization of images," *IEEE Transactions on Communications*, vol. COM-33, pp. 1208–1219, November 1985.

[57] R. Arvind and A. Gersho, "Image compression based on vector quantization with memory," *Optical Engineering*, pp. 570–580, July 1987.

[58] L. Yin, "Performance comparisons of LSP coding algorithms based on predictive vector quantization," *Proc. ICSPAT .'95, Boston, USA*, pp. 1878–1882, 1995.

[59] P. Chang and R. M. Gray, "Gradient algorithms for designing predictive vector quantizers," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-34, pp. 679–690, August 1986.

[60] R. A. Wiggins and E. A. Robbinson, "Recursive solution to the multichannel filtering problem," *Journal of Geophysics Research*, vol. 70, pp. 1885–1891, 1965.

[61] D. Manolakis, V. K. Ingle, and S. M. Kogen, *Statistical and Adaptive Signal Processing.* McGraw-Hill, 2000.

[62] J. G. Proakis, *Digital Communications.* McGraw-Hill, 3 ed., 1992.

[63] N. Gaarder and D. Slepian, "On optimal finite-state digital transmission systems," *IEEE Transactions on Information Theory*, vol. IT-28, pp. 167–186, March 1982.

[64] S. Benedetto, E. Biglieri, and R. Daffara, "Modeling and performance evaluation of nonlinear satellite links- a Volterra series approach," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-15, pp. 494–506, July 1979.

[65] S. Benedetto and E. Biglieri, "Nonlinear equalization of digital satellite channels," *IEEE Transactions on Special Areas In Communications*, vol. SAC-1, pp. 57–62, January 1983.

[66] S. Chen, J. B. Gibson, C. F. N. Cowan, and P. M. Grant, "Adaptive equalization of finite non-linear channels using multilayer perceptrons," *Signal Processing*, vol. 20, pp. 107–119, 1990.

[67] S. Chen, G. J. Gibson, C. F. N. Cowan, and P. M. Grant, "Reconstruction of binary signals using an adaptive radial-basis function equalizer," *Signal Processing*, vol. 22, pp. 77–93, 1991.

[68] P. Yahampath and M. Pawlak, "Fixed-order decoding for vector quantization over noisy channels," in *Proc. IEEE 2000 Canadian Conference on Electrical and Computer Engineering*, pp. 68–72, May 2000.

[69] P. Yahampath and M. Pawlak, "Soft decoding in noisy channel vector quantization using a neural-net receiver," in *Proc. IEEE International Symposium on Intelligent Systems for Communications and Signal Processing*, November 2001.

[70] V. A. Vaishampayan, *Combined Source-Channel Coding for Bandlimited Waveform Channels.* Ph.D. Thesis, University of Maryland College Park, MD USA, 1989.

[71] W. Härdle, *Applied Nonparametric Regression.* Cambridge University Press, 1990.

[72] V. Cherkassky and F. Mulier, *Learning From Data.* John Wiley, 1998.

[73] B. Aazhang, B.-P. Paris, and G. C. Orsak, "Neural networks for multiuser detection in code-division multiple-access communications," *IEEE Transactions on Communications*, vol. 40, pp. 1212–1222, July 1992.

[74] G. Cybenko, "Approximation by superposition of sigmoidal functions," *Math. Control, Signal, Systems*, vol. 2, pp. 303–314, 1989.

[75] M. T. Hagan, H. B. Demuth, and M. H. Beale, *Neural Network Design.* Boston, MA: PWS Publishing, 1996.

[76] B. D. Ripley, *Pattern Recognition and Neural Networks.* Cambridge University Press, 1996.

[77] H. Demuth and M. Beale, *Neural Network Toolbox User's Guide (Version 4).* The Mathworks Inc (available at: www.mathworks.c om), 1994-2000.

[78] J. Lim and D. Neuhoff, "Joint and tandem source-channel coding with delay constraints," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. SPCOM–P6.8, May 2001.