INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

Université d'Ottawa • University of Ottawa

Soft Trellis Waveform Compression and Joint Source/Channel Coding

by

Tariq F. Haddad, B.Sc.E., M.Sc.E.

A thesis submitted to the
School of Graduate Studies and Research
of the University of Ottawa in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy

Ottawa-Carleton Institute for Electrical and Computer Engineering
School of Information Technology and Engineering
Faculty of Engineering
University of Ottawa

Ottawa, Ontario, Canada
October 2000

Abstract

This thesis aims at improving the performance of trellis waveform compression systems and joint source/channel coders. We first provide the theoretical framework for the new concept of soft source coding, which is based on exploiting the similarity between trellis source compression and maximum likelihood channel decoding. The new approach involves using the maximum *a posteriori* soft-output channel decoding algorithm for trellis source compression. Given a block of source output vectors, the new algorithm delivers a set of soft reliability values that describe the likelihood of the different symbols at the encoder output, in the minimum distortion sense. As a trellis search algorithm, the new soft procedure performs as well as the Viterbi algorithm.

The soft reliability information delivered during compression is employed to derive a fuzzy trellis codebook search algorithm. The new algorithm outperforms the Linde-Buzo-Gray (LBG) algorithm in delivering lower distortion reproduction codebooks. Unlike the LBG algorithm, the new fuzzy approach is significantly less sensitive to the initialization phase, and provides improved performance using "short" training sequences. This feature is particularly of interest with practical nonstationary sources of information. The success of this fuzzy algorithm is extended to the case of channel-optimized trellis waveform compression. Over a practical range of channel bit error rates, and compared to the channel-optimized LBG algorithm, the new approach provides lower distortion channel-optimized codebooks irrespective of initialization using short training sequences. Furthermore, the optimal channel-optimized codebooks show improved performance under channel-mismatch conditions.

The last part of this thesis addresses the design of joint source/channel (JSC) soft decision decoding/detection systems. First, we study the performance of soft decision JSC decoding using k-DPCM compressed images over memoryless Gaussian channels with convolutional channel codes. Then, we show the advantage of using higher redundancy levels in providing higher coding gains, especially during "bad" channel conditions. This advantage is highly utilized when the alphabets of the source encoder output match the channel encoder input. For this purpose we test the performance using dual-k convolutional codes and k-DPCM systems. Furthermore, we provide a comparative study between sequence-MAP and symbol-MAP JSC detection. Although the two approaches provide similar bit error rates, we show that symbol-MAP detection is always associated with lower average distortions of reproduction as applied with VQ systems.

To them who know me imperfect and love me

My Parents

Acknowledgments

I would like to express my sincere gratitude to my supervisor Prof. Abbas Yongaçoğlu for his constructive guidance and support during my period of study. I would like to thank the committee members Jean-Yves Chouinard, Claude D'Amours and Mohamed El-Tanany for their valuable comments, which improved the quality of this thesis research. Finally, I wish to thank my friend Colleen Ennett for always being there for me.

Contents

List of Figures

List of Tables

List of Acronyms

APP	A *Posteriori* Probability
AR	Auto-Regressive
AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
BCJR	Authors last names (Bahl, Cocke, Jelinek, Raviv)
BSC	Binary Symmetric Channel
bpp	Bits per Pixel
bps	Bits per Sample
BPSK	Binary Phase Shift Keying
CI	Confidence Interval
CO-FSTVQ	Channel-Optimized Fuzzy Soft Trellis Vector Quantization
CO-LBG	Channel Optimized Linde-Buzo-Gray Algorithm
CO-STVQ	Channel-Optimized Soft Trellis Vector Quantization
CO-TVQ	Channel-Optimized Trellis Vector Quantization
COVQ	Channel-Optimized Vector Quantization
CO-VTVQ	Channel-Optimized Viterbi Trellis Vector Quantization
DA	Deterministic Annealing
FSM	Finite State Machine
FSTVQ	Fuzzy Soft Trellis Vector Quantization
IA	Index Assignment
IIR	Infinite Impulse Response
JSC	Joint Source/Channel
LBG	Linde-Buzo-Gray Algorithm
LLR	Log-Likelihood Ratio
MAP	Maximum A Posteriori Probability
MDTA	Minimum Distorion Theoretically Attainable
ML	Maximum Likelihood
NNC	Nearest Neighbour Condition
OPTA	Optimal performance Theoretically Attainable

PCM	Pulse Code Modulation
PSNR	Peak Signal to Noise Ratio
RCE	Recursive Convolutional Encoder
SA	Simulated Annealing
SISO	Soft-Input/Soft-Output
SM-TVQ	Source-Matched Trellis Vector Quantization
SNR	Signal to Noise Ratio
STVQ	Soft Trellis Vector Quantization
SVQ	Soft Vector Quantization
TCVQ	Trellis Code Vector Quantization
TVQ	Trellis Vector Quantization
VQ	Vector Quantization
VTVQ	Viterbi Trellis Vector Quantization

List of Symbols and Variables

$\mathcal{A}$	Space of the source encoder output alphabet
$\mathcal{C}$	Codebook
C or c	Codeword.
G	Channel capacity.
D	Average distortion
$D(R)$	Distortion rate function
$\bar{D}(R)$	Minimum distortion theoretically attainable
$E\{\cdot\}$	Statistical average
$\mathcal{F}^c(\cdot)$	Error function complementary
$H(\cdot)$	Entropy
h_d	Hamming distance
$h_G(\cdot)$	Differential entropy of the Gaussian source
$H_\infty(\cdot)$	Entropy rate
$I(\cdot,\cdot)$	Mutual information
k	Number of bits to represent a trellis branch index
K	Codebook Cardinality
L	Vector dimension
M_n^m	Channel cumulative decoding metric at state $s_n = m$
n	Time index
$\mathcal{P}$	Coding partition space
$P_{j\|i}$	Channel transition probability between the received symbol j and the transmitte
R	Encoding rate
$R(D)$	Rate distortion function
$\bar{R}(D)$	Optimal performance theoretically attainable
S_i	Coding set
s_n	Trellis state at time n
$S_{yy}(e^{jw})$	Source power spectral density
x_n	Source encoder output symbol
Y_n	Source output vector or channel output vector

$\alpha_n^i(m)$	Forward probability variable for branch i of state m at time n
$\beta_n^i(m)$	Backward probability variable for branch i of state m at time n
$\gamma_n^i(m)$	Probability of Y_n given the codeword colouring branch i out of state m
γ_y^2	Spectral flatness
ϵ	Channel BER
ϵ_a	Actual channel BER
ϵ_d	Design channel BER
ζ	Relaxation time constant
η_y^2	Minimum prediction error variance
ν	Trellis constraint length
$\pi_{l\|j}$	Conditional encoding rule of a vector j into set l
$\pi_n(l\|Y_1^N)$	Conditional encoding rule of a reproduction l given the sequence Y_1^N
ρ_D	Redundancy due to source distribution
ρ_M	Redundancy due to source memory
ρ_T	Total redundancy
σ_n^2	Channel noise variance
σ_y^2	Source variance
τ, κ, η	Lagrangian multipliers
$\Phi(\cdot)$	The function of the finite state machine
$\varphi_n(j)$	Decoding probability at time n of reproduction j
$\psi_n(l)$	Probability of a codeword l at time n given the sequence Y_1^N
$\Psi(\cdot)$	Function denotes the operation of the trellis search algorithm
$\Omega(\cdot)$	Next state function

Chapter 1

Introduction

Efficiency and reliability of data transmission are the two basic fundamentals of digital communications design. In a properly designed system, these two basic concepts map into a balanced trade-off between fidelity of reproduction and communication resources. The required balance is achieved in accordance with the bit error rate, transmission rate, transmission power, channel bandwidth and system complexity. Maintaining both objectives is achieved through redundancy management, which is performed by the different coding systems within the communications link. It is known that higher redundancy levels tend to drive the data stream to become less sensitive to channel errors [1], however, at the expense of increased transmission rates and delay. Accordingly, the redundancy level controls the efficiency of data transmission, which is directly related to the transmission rate. The reliability, on the other hand, is measured by the rate of correctly received data symbols, which is associated with the quality of reproduced information at the destination.

Although some redundancy is augmented into the transmitted data by channel coding, the efficiency of transmission is mainly controlled by source coding. A source coding system typically involves the two operations of data compression and binary coding. The data compression part reduces the level of redundancy inherent in the information sequence. However, at the expense of a loss in the fidelity of reproduction (distortion). In order to further reduce the amount of information necessary for transmission, binary coding is used to mainly apply an efficient codeword assignment technique. The branch of information theory that studies the relation between compression rate and distortion is the rate-distortion theory [2]. This theory is devoted to characterizing the distortion of reproduction at rates lower than

the average amounts of information (entropy) provided by the source output. One of the fundamental outcomes of the rate distortion theory is that irrespective of the source statistics, improved performance can be always achieved by block (vector) processing of the source output rather than scalar processing. This basic result motivated research and development in the field of vector quantization and its variants [3]. In chapter 2 we introduce the basic concepts of vector quantization (VQ) as well as trellis vector quantization (TVQ), which are used in subsequent chapters for further development. As indicated by the title, the research presented in this thesis is related to two main themes, data compression and joint source/channel coding. In the following few paragraphs we shall formulate the motivation and the problem statements of the material introduced in this thesis.

In [4], the author clearly states the fundamental consepts of maximum likelihood (ML) and maximum a posteriori probability (MAP) detection/estimation. The choice between the two approaches of ML or MAP is strictly related to the statistical characteristics of the channel input sequence. Unlike ML decoding, which is associated with independent and identically distributed (iid) channel input, MAP decoding covers the case of treating correlated and/or nonuniform channel input symbols. Accordingly, ML decoding is a special case of MAP decoding. Both ML and MAP detection can be applied either to minimize the probability of symbols in error or to minimize the probability of sequences of symbols in error (words). Operating on an iid channel input sequence, not only ML is equivalent to MAP, but also minimizing the probability of symbols in error is equivalent to minimizing the probability of words in error. One algorithm that applies sequence-MAP, thus, sequence-ML, decoding is the Viterbi algorithm [5]. Symbol-MAP (or ML) decoding is applied using a forward/backward algorithm, which is referred to as the symbol-MAP algorithm in the area of channel decoding [6]. For each time instant in the decoding window, this algorithm delivers a set of soft reliability information that describes the likelihood of each possible output symbol. Based on the former argument, both the Viterbi algorithm and the symbol-MAP algorithm are expected to perform within the same proximity under iid channel input conditions.

A basic contribution of this thesis represents introducing the forward/backward symbol-MAP algorithm for data compression, in particular for trellis source coding. This novel contribution is formulated based on an analogy drawn between the two topics of channel decoding and source compression. It is known that both areas of research employ the concepts of signal space and Euclidean distance, as well as block and sequential trellis coding. The theory of trellis vector quantization (TVQ) has been well established in the seventies and early eighties [7], [8], [9], [10], [11], [12] and [13]. Further improvements were achieved when the authors in [14] used the fundamental concepts of trellis coded modulation and introduced the "natural

dual" trellis coded quantization (TCQ). The Viterbi algorithm is used with TVQ to search for the minimum distortion path. In chapter 4, we show how TVQ is similar to trellis decoding of iid channel input sequence, which allows us to introduce the symbol-MAP channel decoding algorithm as a trellis source compression algorithm.

The development of chapter 4 includes two parts: algorithm derivation and testing. The testing phase shows the feasibility of using the symbol-MAP algorithm for data compression. Compared to the Viterbi algorithm, the symbol-MAP algorithm provided similar average performance using memoryless Gaussian as well as Gauss-Markov sources. In the context of TVQ, the symbol-MAP algorithm delivers a set of reliability information that describes the likelihood of the different output symbols in the minimum distortion sense. Accordingly it is referred to as the soft trellis vector quantization algorithm (STVQ).

A key-part of a TVQ system is the codebook design phase. Unless the approach is on-line following some adaptive technique [15], the search process is usually iterative, performed using a "long" training sequence. One algorithm that performs off-line codebook design is the LBG algorithm, which represents the practical implementation of the theory of optimal quantization developed in [16] as method I. A heuristic description of this algorithm as applied with VQ and TVQ is introduced in chapter 2. The process of searching for an optimal codebook for a TVQ has several degrees of freedom. These variants are associated with the source statistics, the training sequence length, the distortion surface as well as the trellis structure itself. The procedure of the LBG algorithm is not usually capable of accommodating for these different parameters. This usually leads the algorithm towards descending to locally suboptimal minimum distortion configurations (codebooks). In particular, according to the theoretical framework of the LBG algorithm, the algorithm procedure requires a relatively long training sequence to approximate the source statistical distribution. Practically this condition significantly reduces the efficiency of the developed codebooks, since very often we deal with nonstationary sources of information. Furthermore, since the LBG algorithm is locally optimal, its performance is highly dependent on the initialization phase.

Motivated by solving the problems associated with applying the LBG algorithm for TVQ, we introduce in the second part of chapter 4 a fuzzy codebook search approach using the soft information provided by the STVQ algorithm. The new codebook search technique is derived following an information theoretic approach in accordance with the development of Blahut algorithm [17]. The developed procedure minimizes a soft distortion measure computed using the reliability information delivered by an STVQ step. Minimizing the soft performance measure culminates into a set of codeword update equations, which allow the different source output

vectors to contribute to modifying the different encoding sets at each time instant. The contribution of a training vector to a set update is weighted by the distortion-related likelihood value of the associated set centroid. The term likelihood refers to the possibility of a particular set centroid to be the reproduction codeword. This association step explains the reason for using the name fuzzy to denote the new codebook search process. Accordingly, the process is called fuzzy soft trellis vector quantization (FSTVQ). The fuzziness of this procedure is opposed by the hard association step that is performed by the LBG algorithm. We shall demonstrate through several simulation cases that the new codebook search approach arrives at lower distortion configurations while being significantly less sensitive to the initialization phase as well as to the length of the training sequence. Simulation cases represent the compression of a mixture Gaussian source, a Gauss-Markov source, as well as still images.

The second part of this thesis considers topics in joint source and channel (JSC) coding. Significant advances in the research and development of information and communications theory are based on the separate design of the source and the channel coding systems. For a comprehensive reference to several milestone publications, the reader may refer to the October 1998 commemorative issue of the IEEE Transactions on Information Theory. Nonetheless, for several reasons, which are presented in later chapters, significant improvement in the performance of the overall system comes from a JSC design with a more efficient use of the available resources. Very often, the concept of JSC coding extends beyond coding the information sequence in one redundancy management block to a rather more general understanding. A JSC coding system can be physically separate, yet it is joint in the sense of considering the statistics of either of the source sequence in the design of the channel coder, or the channel in the design of the source coder. Therefore, a broader understanding of the concept of JSC coding represents the idea of employing the available information at any point through out the communications link in the design of other parts of the overall system.

Several publications in literature consider the joint design of source and channel coding systems. For example we mention the following documented classes of such systems. In [18], [19] and [20], the authors incorporate the soft information delivered by a soft output channel decoder in a channel-optimized vector quantizer design. Another category considers the design of one block of redundancy management that takes care of compression and delivers a channel sequence, which is less vulnerable to transmission errors. This class includes all channel-optimized source compression systems like the ones developed in [21, 22], [23, 24, 25] and [26, 27] along with the attempts in [28], [29] and [30]. In particular, Massey in [28] developed a theorem for distortionless JSC linear coding system. On the other hand, Helman in [29, 30]

studies the possibility of using a simple convolutional encoder to perform compression, and at the same time, error protection using the source natural redundancy. In the last example we mention the class that employs the statistics of the information sequence as a constraint in the design of a channel decoder/detector. In this class, the residual redundancy inherent in the information sequence is represented by a Markov model, which is used in a MAP metric for JSC decoding/detection. Some of the major publications that consider source-optimized channel decoding/detection are [31], [32], [1], [33, 34], and [18, 35]. The information sequence referred to in these publications is at the output of either the source of information or the source encoder.

In the context of the second class of JSC coding systems, chapter 5 treats a channel-optimized TVQ coding system. The first practical application of the theory of channel-optimized TVQ was introduced in [22]. After successfully adapting the LBG algorithm with trellis waveform coding in [12], the same algorithm was the intuitive choice for the authors in [22] as a codebook search algorithm. It is known that the problems associated with the LBG algorithm become vanishingly less pronounced as the channel becomes more noisy. Nevertheless, the algorithm is still very sensitive to initializations and training sequence lengths over moderate-to-small, yet practical, channel bit error rates. For this reason, we introduce in chapter 5 a modified version of the FSTVQ approach to fit the context of channel-optimized TVQ. Before developing the theoretical framework for channel-optimized FSTVQ, we provide in this chapter a simple and comprehensive introduction to channel optimized trellis waveform coding. This introduction is presented in relation to the more general problem of channel-optimized vector quantization. Several simulation cases are introduced to demonstrate the potential of channel-optimized FSTVQ in providing lower distortion channel-optimized codebooks that are reasonably robust to channel mismatch conditions. Furthermore, we show in this chapter the significance of channel optimized TVQ systems compared to tandem systems in terms of saving extra delay, complexity and bandwidth requirements.

In chapter 6 as well as in appendix E, we apply and investigate the work established for source-constrained channel decoding/detection systems. In this class the residual redundancy at the source encoder output is used to enhance the probability of correct decision at the output of the channel decoder/detector. We first extend the existing theory, [34, 33, 18, 35], to accommodate for soft decision decoding, which as expected provided higher JSC decoding/detection gain. Then, through several simulation scenarios we show the impact of using higher redundancy levels on improving the performance of source-optimized channel decoding. The importance of matching the source encoder output alphabet to the channel encoder input is shown in [33] through using nonbinary convolutional codes. In chapter 4, we fur-

ther investigate this point using dual-k nonbinary convolutional codes with k-DPCM compressed images. The authors in [18] and [35] develop a source-optimized channel detection system using sequence-MAP Viterbi detection and a new instantaneous-MAP algorithm. In Appendix E, we provide a simple comparison study between sequence-MAP, instantaneous-MAP, and symbol-MAP soft as well as hard detection. We shall also show the advantage of using JSC symbol-MAP detection in providing lower average distortions of reproduction with VQ-based compression techniques.

1.1 Thesis Outline

This thesis is arranged into seven chapters including the introduction and the conclusions chapters. The presentation of the five intermediate chapters unfolds as follows:

- **Chapter 2** introduces a comprehensive background on the topics of rate distortion theory, vector quantization and trellis waveform coding. Furthermore, we introduce in this chapter the concept of deterministic annealing, which is related to the development of the FSTVQ approach in chapter 4. Although the presentation is supported by the necessary theoretical background, it can be easily followed for programming purposes.

- **Chapter 3** considers the essentials and the advantages of soft channel decoding as an introduction for the presentation of the symbol-MAP algorithm. Once again, the presentation format along with the explanatory figures facilitate employing the algorithm for different applications.

- **Chapter 4** represents the core contribution of this thesis. In this chapter we present the source/channel analogy followed by the development of the STVQ algorithm. Then we talk about the complexity of this algorithm in relation to the Viterbi algorithm. After that, a comparative study shows the close performance of both the STVQ algorithm and the Viterbi algorithm as search procedures for the minimum distortion path on the encoding trellis. In the second half of this chapter we extend the STVQ algorithm to provide a reliable and robust codebook search approach, which is referred to as the FSTVQ algorithm (or approach).

- **Chapter 5** first provides an easy to follow, yet comprehensive introduction to the topic of channel-optimized TVQ in relation to COVQ. Then, the chapter deals with extending the FSTVQ approach to become a codebook search process over noisy channel. As we shall demonstrate, the new developed procedure provides significant improvement over the LBG algorithm in the sense of

delivering lower distortion codebooks. The improvement in performance is accompanied by being less sensitive to the initialization phase using a relatively short training sequence over practical noisy channels.

- **Chapter 6** investigates and extends the existing source-constrained channel decoding theory. The development shows the advantage of using soft decision JSC decoding along with significance of higher residual redundancy levels. Furthermore, through the different considered simulation cases we show the advantage of using source-matched JSC decoding.

The last part of this thesis includes five appendices that serve to illustrate several topics, which are mentioned through out this thesis.

1.2 Thesis Contributions

The main objective of design in communications is to enhance the system performance towards the classical theoretical limits, which are represented by the concepts of channel capacity and rate distortion function. Several break throughs in the area of channel coding have led to major advancements towards reaching the channel capacity limit. For example it has been shown recently that turbo channel codes , [36, 37], can perform within 0.6 dB of the theoretical capacity limit. Motivated by the performance of some channel coding systems, researchers have considered reflecting the same ideas into the area of source coding, [28], [29] and [14]. Similarly, I started my work on this thesis research inspired by the power of soft output channel decoding algorithms, especially with iterative decoding [38, 39]. The work established in this thesis can be considered as a step towards introducing the concept of soft output channel decoding for source compression. The following represents the major contributions of this thesis

1. Applying the symbol-MAP algorithm for trellis waveform coding.
2. Developing a new reliable and robust trellis codebook design approach using the distortion-related reliability information, which is delivered by the soft compression algorithm.
3. Extending the new codebook design approach to the problem of channel-optimized TVQ.
4. Modifying the existing metric of source-constrained channel decoding for soft decoding. Besides, showing the advantage of using dual-k convolutional codes with k-DPCM systems and joint source/channel decoding.

Chapter 2

Lossy Data Compression

Despite the fact that lossless data compression techniques guarantee the exact reproduction of information sources, the rate of compression is always lower bounded by the source entropy. The source entropy represents the amount of information (uncertainty) inherent in the source output symbols. Therefore, if we need to preserve the information at the source output (lossless compression), we have to make sure not to cross the entropy lower limit. However, the storage and the channel capacity limitations of communication systems usually demand further lower rates of compression. For such applications, where it is more affordable to lose information as opposed to providing more expensive communications resources, we consider lossy source compression techniques.

The fidelity of reproduction at the destination is specified by an average distortion measure D, which constraints the minimum rate information sequence. The minimum rate of information, which is associated with D, is the rate distortion function $R(D)$. This limit specifies the minimum rate of compression required to reproduce the source of information with average distortion D. In other words, compressing at rates lower than $R(D)$ would result in average distortions higher than D. The concept of the rate distortion function was first introduced by Shannon in 1959 [40], and it is very similar to the measure of channel capacity. In fact, while the channel capacity represents the maximum amount of mutual information that can be transmitted over a channel given the suitable distribution, the rate distortion function represents the minimum amount of mutual information between the source output and its reproduction over the encoding-decoding channel constrained by a distribution and the allowed distortion. It is clear that there is a relationship between

Table 2.1: The distortion rate values of a pdf-optimized uniform quantizer (pdf-OUQ), a vector quantizer (VQ) and a trellis vector quantizer (TVQ) for a memoryless Gaussian source.

	D(R)			
R *(bits/sample)*	pdf-OUQ	VQ	TVQ	Theoretical
0.5	–	0.5734	0.5110	0.50
1	0.3631	0.3318	0.2836	0.25
2	0.1189	0.1081	0.0853	0.0625
3	0.0374	0.0311	0.0252	0.0156

the rate distortion function and source entropy, which represents the amount of redundancy-free information produced by a source. According to the source coding theorem, the source entropy sets the lower limit for lossless source compression and compaction. Thus, on the rate distortion curve, the source entropy represents the value of the rate distortion function associated with perfect reconstruction; i.e., $R(0)$. Accordingly, for compression rates $R(D) < R(0)$, we actually go beyond redundancy removal to the point of dropping information, which we cannot reconstruct.

From a system design point of view, the two parameters that limit the performance of a communication system are the channel capacity and the rate distortion function. It is essential to know the conditions under which it is possible to convey information without exceeding some prespecified upper average distortion limit D. It is known that for reliable communications, the channel capacity should be larger than the minimum rate of the information sequence, which is required to reproduce the source at a given fidelity criterion. Therefore, given a source of information, a channel and a constraint D, the designer should be able to control the channel capacity G (change the energy per channel symbol) in order to satisfy $R(D) < G$. The other way is correct too, given a channel capacity G, which is mainly controlled by the modulation technique and the channel itself, chose the proper operational $R(D)$ that meets $R(D) < G$. Accordingly, the knowledge of the rate distortion function, whenever available, is of significant importance for the design of communication systems.

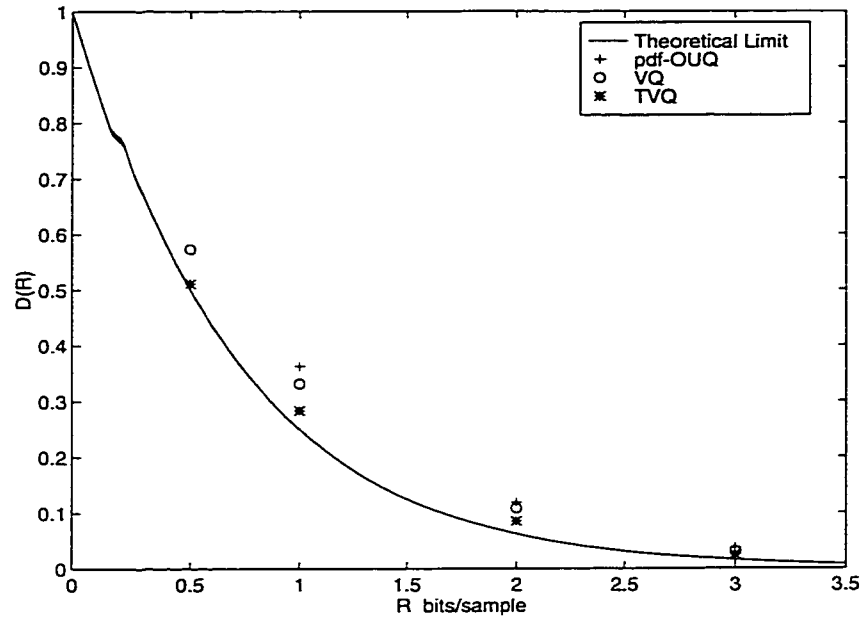

Figure 2.1: The MSE distortion rate function of a Gaussian source with the performance of a pdf-optimized uniform quantizer (pdf-OUQ), a vector quantizer (VQ) and a trellis vector quantizer (TVQ).

The other measure of interest is the distortion rate function $D(R)$. This function provides us with the lowest possible limit of average distortion that can be achieved at a given compression rate. The distortion rate function is used to compare the performance of different data compression techniques. The comparison is based on how close the average distortion of reproduction is to $D(R)$ at a given information rate R. As an example, the mean-squared distortion rate function of a memoryless Gaussian source is given as [2]

$$D_G(R) = 2^{-2R}\sigma_y^2 \tag{2.1}$$

where R is in bits/sample and σ_y^2 is the source variance. Table 2.1 and Fig. 2.1 show the performance of three lossy compression techniques compared to the theoretical distortion rate limit for an independent and identically distributed Gaussian source. The performance results of the pdf-optimized uniform quantizer is borrowed from [41]. However, we evaluated the performance of both vector quantization (VQ) and trellis vector quantization (TVQ) by using a training sequence of 30000 samples with different constraint-lengths and dimensions to achieve the specified compression rates.

This chapter introduces the basic concepts of vector quantization as well as trellis vector quantization. These two approaches are block lossy source encoding techniques, which map the source output vectors into an encoding space represented by a set of reproduction codewords. Furthermore, we shall present the subject of deterministic annealing, which is used for data clustering. These different topics are used later in this thesis for further developments.

2.1 Vector Quantization

As recognized by Gray in [42], rounding off is the earliest forms of quantization, which goes back to 1898. However, PCM is considered the most advanced origins of quantization. This technique was first adopted in 1938 as a quantization/modulation approach [43], [42]. In 1957 Lloyd established two methods, one is variational and the other is iterative, to search for a locally optimal scalar quantizer. His results were first published in a special issue on quantization in 1982 [16]. The advantage of using block source coding approaches over scalar coding in the rate-distortion sense was clearly verified in the classical source coding theories [2, 44, 45, 46, 40, 47]. Consequently, it was intuitive to extend the basic design algorithms for scalar quantizers to accommodate for multidimensional vector quantization. Vector quantization is particularly important in applications where the source of information exhibits certain levels of correlation depth (dependency between successive samples), such as speech and image sequences.

From an operational point of view, VQ involves first mapping a source output vector into an encoding space specified by a source-matched reproduction codebook. The quantizer, then, generates the index that points to the codeword, which is closest in a prespecified distortion sense. The source decoder, at the other end, operates using the same encoding codebook to produce the proper reproduction codeword that corresponds to the received index. The decoding process is performed using a table lookup technique, as shown in Fig. 2.2. It is clear that although the encoding process requires a considerable number of comparisons and computations, the decoding process is fairly simple. Several implementation and design issues of vector quantization are heuristically presented in [3].

To mathematically describe VQ, we consider a discrete-time continuous-amplitude source $\{y_n\}_{n=0,1,2,\ldots}$ blocked into L-dimensional vectors as

$$Y_n = [y_{nL}, y_{nL+1}, \ldots, y_{nL+L-1}]$$

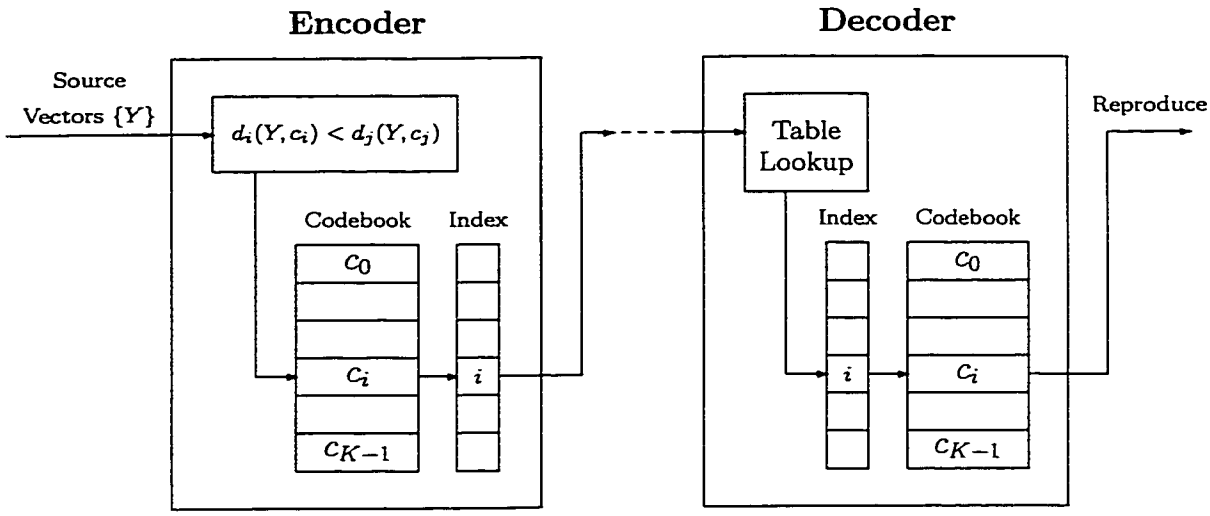

Figure 2.2: Vector quantization.

An L-dimensional VQ, with a codebook $\mathcal{C} = \{c_0, c_2, ..., c_{K-1}\}$, encodes the source output vectors at a rate of $R = \log_2(K)/L$ bits/sample (bps). The value of the codebook cardinality K at a given average distortion D and for sufficiently large L is theoretically computed using [2]

$$K(D, L) = 2^{\{L \cdot R(D)\}}$$

A vector-space treatment of the structure of VQ is presented in [48]. In that paper Gersho provides a proof for the necessary conditions of the existence of a locally optimal VQ. The first intuitive condition is the Nearest Neighbor Condition (NNC), which represents the VQ encoding rule. This rule guarantees minimizing the overall average Euclidean distance (for the squared-error distortion case) between the information vectors and the corresponding partition set representatives. Given the VQ codebook, optimal clustering partitions the encoding space into K-different sets $\mathcal{P} = \{S_0, S_2, ..., S_{K-1}\}$ such that

$$S_i = \{Y_n : d_i(Y_n, c_i) < d_j(Y_n, c_j) \quad \forall j \neq i; i, j = 0, 1, ..., K-1\} \tag{2.2}$$

where for the squared-error distortion measure, $d_i(Y_n, c_j)$ is given by

$$d_i(Y_n, c_i) = \sum_{l=0}^{L-1} (Y_{nL+l} - c_{il})^2 \tag{2.3}$$

As depicted by Eq. (2.2), a partition set S_i contains all the source output vectors that are represented by the codeword c_i.

The other condition of optimality controls our choice of the VQ codebook $\mathcal{C}$, which is derived by minimizing the average distortion measure given the partition $\mathcal{P}$. The average distortion is given by

$$D_Q = \frac{1}{L} \sum_{i=0}^{K-1} \int_{S_i} p(Y) d_i(Y, c_i) dY \tag{2.4}$$

where $p(Y)$ is the joint probability density function of the vector Y. The minimization is performed by applying the orthogonality principle of linear estimation [48]. Accordingly, the set of optimal reproduction codevectors, which produces the minimum average distortion, is the conditional mean

$$c_i = E\{Y | Y \in S_i\} \quad i = 0, 2, \ldots, K-1 \tag{2.5}$$

where the expectation is over the distribution of the information vectors. It is known that a particular set exhibits minimum moment of inertia around its centroid. Considering that the average distortion is equivalent to the total moment of inertia of the partition $\mathcal{P}$, an optimal codevector computed using Eq. (2.5) is usually interpreted as the center of mass (centroid) of the corresponding partition set.

The above mentioned VQ optimality conditions are extensions to those of the scalar quantizer, which were developed in 1957 by Lloyd [16]. The algorithm that uses these conditions for VQ codebook design is the Linde-Buzo-Gray (LBG) algorithm [49], which is also known as the Generalized-Lloyd algorithm. This iterative algorithm uses a long training sequence that approximates the source distribution and guarantees the convergence to a locally optimal average distortion point in a nonincreasing manner. The procedure starts with an initial codebook, generated using one of the techniques mentioned in [50] or in [51]. Since the algorithm is locally optimal, it utterly depends on the choice of the initial codebook. In this thesis we use the splitting technique described in [49] to initiate the codebook search process. In the second step the algorithm performs NN-encoding and forms the partition $\mathcal{P}$ over the given encoding space. The reproduction codewords, that constitute the encoding codebook, are then updated by the centroids of the corresponding partition sets. These steps are repeated until a minimum average distortion is achieved according to Table 2.2. It can be shown that in the case of squared-error distortion the set centroid is simply the average value of the elements of that particular set

$$c_{ij}^{(m+1)} = \frac{1}{||S_i^{(m)}||} \sum_{l \in S_i^{(m)}} Y_{lj}$$

Table 2.2: The LBG Algorithm

Step 1: Initialize the algorithm $m = 0$ and $D^{(0)} = 0$, ϵ. Start with an initial set of

- Reproduction vectors $\{c_i^{(0)}\}_{i=0}^{K-1}$.

- Training vectors $\{Y_n\}_{n=0,1,\ldots}$.

Step 2: Decide on the quantization regions

- $S_i^{(m)} = \{Y_n : d_i(Y_n, c_i) < d_j(Y_n, c_j) \quad \forall j \neq i\}_{i,j=0,1,\ldots,K-1}$

Step 3: Compute the average distortion $D^{(m)}$ between the training vectors and their reproduction value

- $D^{(m)} = \frac{1}{L} \sum_{i=0}^{K-1} \frac{1}{||S_i^m||} \sum_{j \in S_i^m} d_i(Y_j, c_i)$

Step 4: If $\frac{D^{(m)} - D^{(m-1)}}{D^{(m)}} < \epsilon$ stop, otherwise, continue.

Step 5: $m = m + 1$, update the values of $\{c_i^{(m)}\}_{i=0}^{K-1}$ by the centroids of $\{S_i^{(m-1)}\}_{i=0}^{K-1}$.

Go to Step 2.

2.2 Deterministic Annealing for Data Clustering

The convergence of the LBG algorithm to the global minimum-distortion point is highly dependent on the initial codebook as well as the nature of the distortion surface [52], [53]. It is very likely that a poorly-initialized LBG algorithm gets trapped at a locally optimal minimum-distortion point on a non-convex distortion surface. Several publications considered modifying the search process by introducing a level of randomness in order to avoid local minima [54], [55], [56] and [57]. A global optimization technique that proved to be efficient in the context of source coding is simulated annealing [58], [24], [59] and [60]. The thermal equilibrium of the physical annealing process is a phenomenon with multi degrees of freedom, and it is statistically described in [61]. From a physical point of view, annealing is performed to force a material into lower energy states by gradually reducing its temperature, spending more time around critical temperature points. The corresponding probabilistic annealing algorithm that simulates this physical technique is a Gibbs distribution-based procedure, which assigns high probability values to low-energy configurations

[62]. Although simulated annealing (SA) showed potential in providing lower energy (distortion) configurations, it is a computationally demanding process, which is extremely sensitive to the temperature or the annealing schedule.

On the other hand, deterministic annealing (DA) features the best of the two worlds. It is deterministic in the sense that the algorithm achieves an average gradual progress towards the global minimum of the cost surface without randomly pondering across local minima [56]. The development of deterministic annealing goes back to the late 1980's early 1990's, and it is based on replacing the random search performed by simulated annealing by expectation [63], [64] and [65]. However, the mathematical basis of deterministic annealing is based on the principles of information theory of minimizing a cost function subject to entropy constraints.

Generally speaking, clustering is used for the analysis and processing of signals without *a priori* knowledge of their probabilistic distributions. In particular, clustering in vector quantization involves representing a source output vector Y by the best codevector $C \in \mathcal{C}$ in the minimum distortion sense. In order to introduce DA, we will assume that the source output vectors are assigned to a partition set with an association probability. Accordingly, we define the average distortion as

$$\begin{aligned} D &= \sum_{Y} \sum_{C} P(Y,C)\, d(Y,C) \\ &= \sum_{Y} P(Y) \sum_{C} P(C|Y)\, d(Y,C) \end{aligned} \tag{2.6}$$

where $P(C|Y)$ represents an association probability. Note that when a vector Y is assigned to one particular partition set, the association probability has a zero or one value, and the average distortion in Eq. (2.6) becomes identical to that of hard clustering. Searching for an optimal partition involves minimizing D using the free parameters $\{C, P(C|Y)\}$ [56]. However, using these free parameters along with the nearest neighbor clustering rule leads in the limit to a hard clustering suboptimal solution. To prevent that, DA imposes another constraint of randomness in the process of choosing the distribution that minimizes D. The randomness is measured by the joint entropy

$$\begin{aligned} H(Y,C) &= -\sum_{Y} \sum_{C} P(Y,C)\, \log P(Y,C) \\ &= H(C) + H(Y|C) \\ &= H(Y) + H(C|Y) \end{aligned} \tag{2.7}$$

Then, optimization is performed using the Lagrangian

$$J = D - \tau H(Y,C) \tag{2.8}$$

where τ is a Lagrange multiplier. It is clear that as τ increases we trade minimizing the distortion for maximizing the entropy. In the limit as τ approaches zero we arrive at a hard (deterministic) clustering solution. From an information theoretic point of view, maximizing the joint entropy corresponds to minimizing the average amount of information necessary to reproduce the source vectors at a given distortion. We further explain by recalling that the rate distortion function represents the minimum value of the mutual information $I(Y,C)$ over the suitable distribution given an average distortion. However,

$$\begin{aligned} I(Y,C) &= H(Y) - H(Y|C) \\ &= H(Y) + H(C) - H(Y,C) \end{aligned} \tag{2.9}$$

The mutual information quantifies the amount of information that actually passes through the encoding/decoding channel. It follows that in order to minimize $I(Y,C)$, we need to maximize $H(Y|C)$. This means that we can minimize $I(Y,C)$ by maximizing the amount of information about the source output vectors given the reproduction codewords at the decoder output, which is given by the equivocation $H(Y|C)$, or equivalently by $H(Y,C)$.

In order to proceed with the description of DA, we use

$$H(Y,C) = H(Y) + H(C|Y)$$

Note that $H(Y)$ is the source entropy, which is a constant that does not affect the clustering procedure. Besides that

$$H(C|Y) = -\sum_{Y} P(Y) \sum_{C} P(C|Y) \log P(C|Y) \tag{2.10}$$

Then, we write the Lagrangian J as follows

$$J = \sum_{Y} P(Y) \sum_{C} P(C|Y) \{d(Y,C) + \tau \log P(C|Y)\} \tag{2.11}$$

Minimizing J with respect to $P(C|Y)$ results in the Gibbs distribution

$$P(C|Y) = \frac{\exp\left\{-\frac{d(Y,C)}{\tau}\right\}}{Z_Y} \tag{2.12}$$

where

$$Z_Y = \sum_{C} \exp\left\{-\frac{d(Y,C)}{\tau}\right\} \tag{2.13}$$

Minimizing J in Eq. (2.11) with respect to C results in the condition [56]

$$\sum_{Y} P(Y,C)\frac{d}{dC}d(Y,C) = 0 \quad \forall C \in \mathcal{C} \tag{2.14}$$

Eq. (2.14) can be approximated by

$$\sum_{Y} P(C|Y)\frac{d}{dC}d(Y,C) = 0 \quad \forall C \in \mathcal{C} \tag{2.15}$$

which for the squared error distortion leads to the following centroid update equations

$$C = \frac{\sum_Y Y P(C|Y)}{\sum_Y P(C|Y)} \quad \forall C \in \mathcal{C} \tag{2.16}$$

This concludes the derivation of the DA algorithm for clustering or VQ. In summary DA for VQ performs the following main steps:

1. Start with an initial codebook C', and high τ.
2. Compute the association probabilities using Eq. (2.12).
3. Update the codebook using Eq. (2.16).
4. Compute J, reduce τ, and iterate until τ approaches zero, at which point we minimize D and perform hard clustering.

Clearly, the algorithm procedure is identical to that of the LBG algorithm, especially at $\tau = 0$. Clustering the source output vectors $\{Y\}$ into the different partition sets starts with a very fuzzy association rule (high level of randomness) at high τ. The process slowly adopts a hard clustering (nearest neighbor) rule as τ approaches zero. Note that at high "temperatures" (τ) the DA process is extremely fuzzy, thereby, the training vectors influence the update of the different partition sets. Accordingly, the iterative DA relaxation process provides the proper initial codebook, which is matched to the training set, to start the hard clustering phase at $\tau = 0$. In the case of hard clustering algorithms, the source output vectors only affect the update of the nearest codeword. This in turn limits the capability of such algorithms to probe upcoming optimal configurations.

2.3 Trellis Vector Quantization

The efficiency and possibility of implementing sequential decoding algorithms of tree-structured error-correcting codes directed research interests into trellis source

coding since the late sixties, [66], [9], [8],[67], [68], [10], [11], [69], [70] and [51]. This section provides a heuristic presentation of design and implementation issues of trellis waveform coding or trellis vector quantization (TVQ). The same concept is approached in literature using different names. Since any implementable encoding tree, generated using a finite-state machine, has a corresponding trellis, tree encoding, mentioned in [8, 71, 66], is basically another version of the same concept. Delayed decision coding [41] and multipath search coding [69] could also be classified as trellis waveform coding. TVQ is further extended in [14], [72] by incorporating Ungerboeck's concept of set partitioning [73] in the design of trellis coded vector quantizers, which have expanded sets of quantization levels.

The structure of a TVQ is defined by a finite-state machine (FSM) decoder and the corresponding encoding trellis. Several papers consider the design of a FSM that is matched to the source of information in an approach that is known as finite state vector quantization (FSVQ) [74], [75], [76], [77], [78] and [79]. In general, the design of a FSVQ involves exploiting the correlation left at a regular VQ output into a finite state space. There are two types of FSVQ, depending on whether the codewords are coloring the states or the transitions between the states, accordingly, FSVQ is either labeled-state or labeled-transition. On the other hand, it is very common just to use a simple fixed next-state FSM-decoder, which consists of lookup table addressed by a shift register. Fig. 2.3 illustrates the coding process of a labeled-transition TVQ with a shift register FSM-decoder. This system is characterized by a trellis that has $M = 2^{k(\nu-1)}$ states, constraint length ν and 2^k branches stemming out of each state. The branches are colored or labeled by the indices that address the reproduction codebook $\mathcal{C} = \{c_0, c_2, ..., c_{K-1}\}$, where $K = 2^{k\nu}$. Encoding is performed using a trellis search algorithm, by first deciding on the minimum distortion path that corresponds to a sequence of source output vectors. Then, the encoder generates the branch indices $\{x_0, x_1, \ldots\}$ that constitute the minimum distortion path. Given that a trellis quantizer is encoding L-dimensional source vectors, and since the branch indices are k-bits each, the compression rate is simply $R = k/L$ bits per sample. At the other end, the decoding process involves feeding the encoder output sequence $\{x_0, x_1, \ldots\}$ into the FSM to generate the corresponding codebook indices.

In the following, we provide the notation necessary to describe a trellis coding process. It is assumed that the discrete-time source output sequence is blocked into L-sample vectors $Y_n = [y_{nL}, y_{nL+1}, \ldots, y_{nL+L-1}]$. Then, at time n the trellis encoder produces an output symbol x_n according to

$$x_n = \Psi(s_n, Y_n); \qquad x_n \in \{0, 1, \ldots, 2^k - 1\} \tag{2.17}$$

where $s_n \in \{s^0, \ldots, s^{M-1}\}$ is the encoder state at time n, and $\Psi(\cdot)$ is used to denote the function of the trellis search algorithm. After that, the encoder updates to the

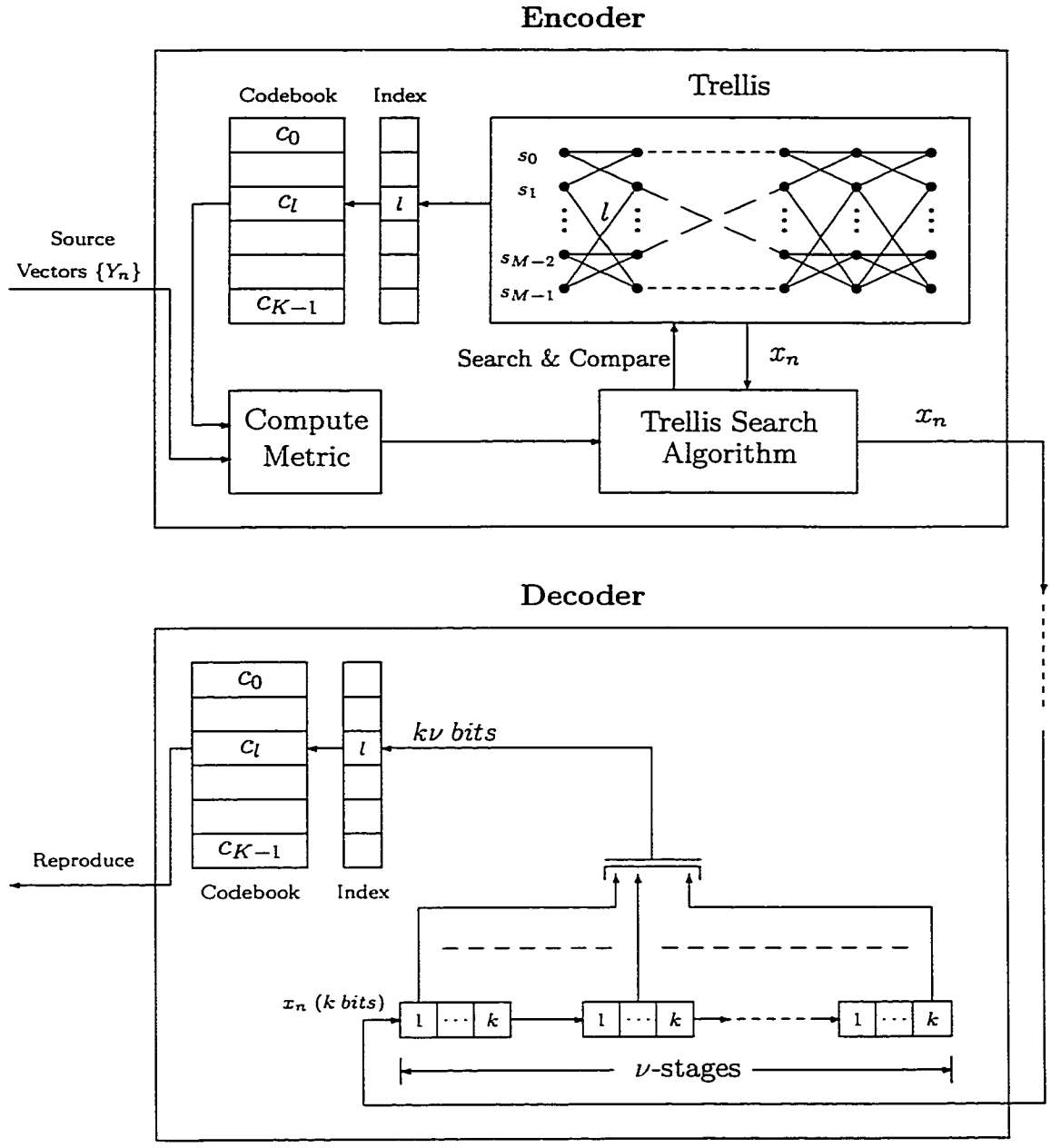

Figure 2.3: Trellis vector quantization

next state following the trellis structure, this operation is modeled by

$$s_{n+1} = \Omega(s_n, x_n) \tag{2.18}$$

On the other hand, decoding at time n involves passing the symbol x_n into the finite state machine of state s_n, which in turn produces the corresponding reproduction codeword using the table lookup technique

$$\hat{Y}_n = \Phi(s_n, x_n); \qquad \hat{Y}_n \in \{c_0, c_1, \ldots, c_{K-1}\} \tag{2.19}$$

Then the decoder changes state according to Eq. (2.18). The employed trellis search algorithm decides on the encoder output sequence subject to the constraint of minimizing an average distortion measure D_Q. For a source output sequence of length N, and squared-error distortion

$$D_Q = \frac{1}{NL}\sum_{n=0}^{N-1} d_n(Y_n, \hat{Y}_n) = \frac{1}{NL}\sum_{n=0}^{N-1} d_n(Y_n, \Phi(s_n, x_n)) \tag{2.20}$$

where

$$d_n(Y_n, \hat{Y}_n) = \sum_{l=0}^{L-1}(Y_{nl} - \hat{Y}_{nl})^2 \tag{2.21}$$

It is interesting to note that we could achieve lower distortion levels for the same compression rate by increasing the complexity of the trellis structure. For example, for a fixed value of k we may reduce the average distortion of reproduction by increasing the reproduction codebook size K. This is achieved by increasing the trellis constraint length ν, since $K = 2^{k\nu}$. It is shown in [9] that the computational average distortion exponentially approaches the distortion-rate function $D(R_o)$ in the following manner

$$D - D(R_o) \rightarrow A2^{-Bk(\nu-1)}$$

where A and B are positive constants, which depend on the compression rate R_0.

2.3.1 Trellis Search Algorithms and Codebook Design

Recall that the basic principle of TVQ is to represent a source output sequence of vectors by the branch indices that constitute the minimum distortion path. Consequently, the process of encoding involves searching for the minimum distortion path throughout the trellis. The authors in [70] provide a survey and cost analysis for several search algorithms that could be used in trellis source encoding systems. The different algorithms are categorized as sorting, non-sorting or exhaustive algorithms.

Sorting algorithms try to isolate the path with the best metric based on an ordering technique that is algorithm-dependent. The M-algorithm [80] is classified as a sorting algorithm, which sorts at a given depth all paths and keeps only the best M-paths before any further encoding. The single-stack algorithm [7], on the other hand, is non-sorting, which keeps storing and following the same path as far as its cumulative metric is above a pre-specified design criterion. The most widely used search algorithm, however, is the Viterbi algorithm [5]. To search for the minimum distortion path, the algorithm employs the principles of dynamic programming in depth-limited exhaustive search approach.

To complete the design of a TVQ, given the trellis and the search algorithm, we need to color (label) the trellis branches with the proper reproduction codewords (codevectors). A rather theoretical approach is to produce, at random, the reproduction codebook using a source-matched probability distribution. This approach was introduced in [68, 8], motivated by the theoretical development of trellis source coding in [66, 9]. Random coding was also employed in [11] to search for the compression codebook, however, in this case based on the results of the alphabet-constrained rate distortion theory presented in [81].

The fact that TVQ is an extension to the general problem of VQ suggests implementing the LBG-algorithm, shown in Table 2.1, in searching for optimal reproduction TVQ codebooks. The algorithm is, accordingly, modified in [12] to fit the general problem of trellis source coding. The algorithm is edited and presented in Table 2.3 to fit into the notion of this section. Note that in step 5 of Table 2.3 the codewords are updated in each partition set by the vectors that produce the minimum average distortion in the corresponding set, which is the set centroid. In order to satisfy the merits of the rate distortion theory, it is desirable to generate an optimal reproduction codebook with a rate distortion function matched distribution. However, the source distribution is not always available to evaluate the proper initial codebook distribution which corresponds to the source rate distortion function. Usually, the only available information about the source distribution is the training sequence itself, which represents a statistical sample function. Accordingly, in several situations, a random initial codebook drawn from the training sequence is used to start the codebook search process, [12] and [49]. The other issue that needs to be addressed is the problem of empty cells. Very often, especially at the early iterations of the search procedure, some of the constituent codewords fail to represent any of the vectors in the training sequence. Consequently, producing empty partition sets, which is clearly a problem since the set centroids are used to update the quantization codebook. To solve this issue we choose a codeword that has a better chance to represent source output vectors. Usually we could either use the same old codeword or another randomly perturbed old codeword or even the centroid of the

Table 2.3: The LBG-algorithm for TVQ

Step 1: Initialize algorithm; a threshold $\epsilon > 0$, iteration index $m = 0$.

An M-state decoder of constraint length ν. A training sequence $\{Y_n\}_{n=0,1,...,N-1}$, and an initial codebook $\{C^{(0)}\}$ of cardinality $K = 2^{k\nu}$.

Step 2: Using generation m codebook $\{C^{(m)}\} = \{c_0^{(m)}, c_1^{(m)}, ..., c_{K-1}^{(m)}\}$, find the minimum distortion sequence $\{\hat{Y}_n\}_{n=0,1,...,N-1}$. This encoding induces a partition on the training sequence $\{S_i^{(m)}\}_{i=0,1,...,K-1} = \{n; \hat{Y}_n = c_i^{(m)}\}$.

Step 3: Compute the average distortion $D^{(m)} = \frac{1}{NL} \sum_{j=0}^{N-1} d(Y_n, \hat{Y}_n)$.

Step 4: If $m > 0$ and $\frac{D^{(m)} - D^{(m-1)}}{D^{(m)}} \leq \epsilon$

Then STOP with $\{C^{(m)}\}$ as the optimal codebook.

Otherwise, go to Step 5.

Step 5: Update the codebook to $\{C^{(m+1)}\} = \{c_i^{(m+1)}\}_{i=0,1,...,K-1}$; such that

$c_i^{(m+1)} = c : \sum_{j \in S_i^{(m)}} d(Y_j, c) \leq \sum_{j \in S_i^{(m)}} d(Y_j, \acute{c})$, $\forall \acute{c}$ (centroid). $m = m + 1$.

Go to Step 2.

entire training sequence [12]. The same empty cell problem is encountered in the context of using the LBG algorithm with VQ.

Chapter 3

The Essentials of Soft Output Decoding

In the context of error control coding, soft-output decoding is simply a procedure that provides subsequent systems with likelihood values of the different possible hard decisions. Soft-output decoding algorithms, in general, produce real numbers that represent the probability of error in decoding particular symbols. Several publications have considered the significance of soft over hard decision decoding, amongst these are [82], [83], [84], [85] and [86]. In these publications, the arguments emphasize the benefits of passing likelihood ratios rather than absolute decisions between different decoding stages. To point out the importance of soft decoding, in his paper [87] Viterbi recommends keeping all decision-related information, which might be useful for subsequent decoding stages, until decoding is complete.

A typical concatenated coding system is shown in Fig. 3.1. The interleaver between the inner and outer encoders is used to force the outer channel to be memoryless. The conventional understanding of serially concatenated codes classifies the inner channel coder as an error re-allocation process, which proceeds the final error correction stage performed by the outer coder. We can show that soft inner coders are similar to noise filters in the sense of operating to optimize a conditional pdf in the process of maximizing the output signal to noise ratio. To further explain, assume that the conditional pdf $p(\mathcal{Y}|x)$ is Gaussian, where the soft (likelihood value) $\mathcal{Y}$ and x are shown in Fig. 3.1. It is known that the probability of symbols in error

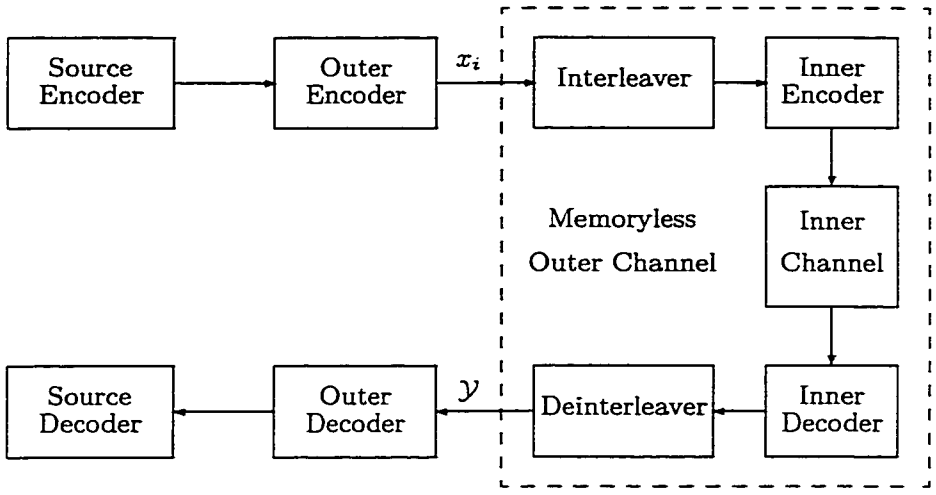

Figure 3.1: A concatenated coding system

at the output of the outer channel is related to the area under the tail of the Gaussian distribution $p(\mathcal{Y}|x)$. Therefore, to reduce the probability of error, the inner soft decoder optimizes $p(\mathcal{Y}|x)$ to minimize the area under its tail. In this particular case of a Gaussian distribution, the area is reduced by minimizing the noise variance over the outer channel. Hence, enhancing the output signal to noise ratio. Intuitively speaking, enhancing the signal to noise ratio increases the maximum amount of information that can be transmitted over the outer channel with reliable recovery. Accordingly, larger outer channel capacity limits are achieved with soft output symbols $\mathcal{Y}$ rather than hard ones.

The capacity of a binary-input $x = \pm 1$ soft-output $\mathcal{Y}$ channel is given by [45]

$$\begin{aligned} G_{soft} &= \max_{P(x_i)} I(X;Y) \\ &= \max_{P(x_i)} \sum_{i=0}^{1} P(x_i) \int_{\mathcal{Y}} p(\mathcal{Y}|x_i) \log_2 \frac{p(\mathcal{Y}|x_i)}{\sum_{j=0}^{1} P(x_j) p(\mathcal{Y}|x_j)} d\mathcal{Y} \end{aligned} \tag{3.1}$$

Since in this equation the logarithm has the base 2, the channel capacity is measured in bits per channel symbol. In the case of symmetry, $p(\mathcal{Y}|-1) = p(-\mathcal{Y}|+1)$, and equally probable input $P(x = \pm 1) = \frac{1}{2}$, Eq. (3.1) reduces to

$$G_{soft} = \int_{\mathcal{Y}} p(\mathcal{Y}|+1) \log_2 \frac{p(\mathcal{Y}|+1)}{p(\mathcal{Y})} d\mathcal{Y} \tag{3.2}$$

where $p(\mathcal{Y}) = \frac{1}{2}\{p(\mathcal{Y}|+1) + p(\mathcal{Y}|-1)\}$. Fig. 3.2 shows the monotonic increase of G from zero to 1 as a function of SNR for an AWGN channel. Note that SNR =

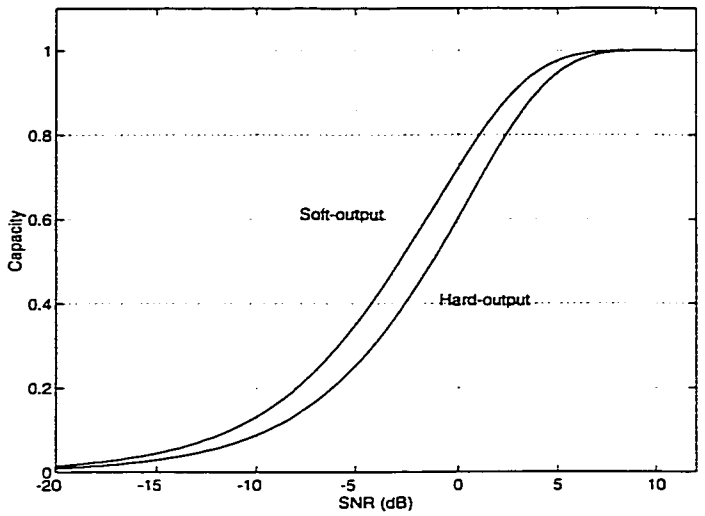

Figure 3.2: Channel capacity in bits per channel symbol

$1/2\sigma_n^2$, where σ_n^2 is the noise variance. On the other hand, the capacity of a binary symmetric channel (BSC) is computed as follows [45]

$$\begin{aligned} G_{hard} &= 1 - H(\epsilon) \\ &= 1 + \epsilon \log_2 \epsilon + (1-\epsilon) \log_2(1-\epsilon) \qquad \text{bits/channel sym} \end{aligned} \tag{3.3}$$

where ϵ is the probability of an incorrect decision, and it is equal to $Q(\sqrt{\frac{2E_s}{N_o}})$, where

$$Q(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}} \exp(-\frac{\lambda^2}{2})\, d\lambda$$

given that $N_o/2$ is the AWGN two-sided spectral height, and E_s is the energy of the channel input symbol. The capacity of the BSC is plotted in Fig. 3.2, where we demonstrate that $G_{soft} > G_{hard}$ for the entire range of SNR.

The original soft output decoding/detection algorithm was developed in [6]. The algorithm minimizes the probability of decoded symbols in error by recursively operating on a block of received noisy data symbols. The algorithm performs MAP-detection/decoding and delivers the *a posteriori* probabilities of all possible output symbols. Different versions of the algorithm are presented in [88], [89], [90], [91], [85] and [92]. The algorithm introduced in [92] is a simplified version of the original [6], which is developed based on the algorithms described in [90] and [91]. On the other hand, the Viterbi algorithm, [5], minimizes the probability of sequences of symbols in error. The algorithm performs MAP or ML (maximum likelihood)

decoding/detection to deliver hard decision symbols. The conventional Viterbi algorithm was first extended to deliver soft-outputs by Hagenauer [93], [84]. The algorithm was modified to produce a reliability information P_c, which represents the probability of a correctly decoded symbol.

In the following sections we introduce two algorithms for symbol-MAP detection and decoding. The theory of soft detection/decoding presented in this chapter is later used in this thesis in a novel approach for data compression.

3.1 Symbol-by-Symbol MAP-Decoding

The fundamental idea of classical hypothesis testing consists of comparing a set of likelihood ratios to another set of hypothesis-dependent thresholds [4]. Computing the likelihood ratios requires estimating the *a posteriori* probability (APP) of a transmitted symbol or sequence of symbols given the received noisy channel output sequence of symbols. In this section we consider the problem of estimating the probability of a transmitted symbol given a channel output sequence of symbols. In general, the channel input sequence is the output of a source of information that exhibits some redundancy at its output. The source of information can be a channel encoder, a source encoder or simply an uncoded sequence of data symbols. As described in appendix B, the redundancy at the channel input is attributed either to the nonuniformity or the correlation between data symbols. Considering that the correlation level at the channel input is described by a Markov process, the problem at hand boils down to estimating the APP of the elements of a channel input Markov process given the output of that noisy channel.

The basic algorithm used to estimate the APP consists of a forward-backward procedure, [94, 95], and it was introduced for the purpose of decoding linear codes in [6]. A Markov process is characterized by a number of states, an alphabet size in each state, and a probability mass function that describes the output alphabets as well as the transitions from one state to another [96]. On the other hand, a Markov process could also be described by a set of conditional probability assignments between the outputs at different time instances. Based on this fact, we shall describe two algorithms with the forward-backward procedure. Given that x_n is the channel input symbol at time n, the first algorithm operates on

$$\{Pr(x_n|x_{n-1}, x_{n-2}, \ldots)\} \tag{3.4}$$

to estimate the APP of the transmitted symbols given the received noisy sequence. This algorithm is rather simple and it is used for MAP-detection, whenever an

estimate of the *a priori* probabilities in Eq. (3.4) is available. Besides that, it provides a simple heuristic approach to describe the forward-backward procedure. The second algorithm, however, estimates the APP of the elements of the Markov process given a received sequence. This algorithm is most useful in applications that involve trellis-structured sequential decoding.

3.1.1 Symbol-MAP detection

Let the input to a discrete memoryless channel at time n be $x_n = a \in \mathcal{A} \triangleq \{a_1, a_2, \ldots, a_J\}$, and the output be y_n. Furthermore, assume that the correlation at the channel input is characterized by a first order Markov model with $\{P_{ij}\} = \{\Pr\{x_n = a_i | x_{n-1} = a_j\}\}$. Then, the optimum symbol-by-symbol MAP detector decides on the symbol according to

$$\hat{x}_n = \arg \max_{x_n \in \mathcal{A}} \Pr\{x_n | Y_1^N\}$$

where N is the length of the observation sequence. Therefore, given a received sequence Y_1^N, we need to find $\Pr\{x_n = a_i | Y_1^N\}; n = 1, 2, \ldots, N, i = 1, 2, \ldots, J$. If we define

$$\lambda_n(a_i) = \Pr\{x_n = a_i, Y_1^N\} \tag{3.5}$$

Then, the measure of interest is computed as follows

$$\Pr\{x_n = a_i | Y_1^N\} = \frac{\lambda_n(a_i)}{\Pr\{Y_1^N\}} = \frac{\lambda_n(a_i)}{\sum_{j=1}^{J} \lambda_n(a_j)} \tag{3.6}$$

Since the channel is memoryless, $\lambda_n(a_i)$ can be written as

$$\begin{aligned} \lambda_n(a_i) &= \Pr\{x_n = a_i, Y_1^n\} \cdot \Pr\{Y_{n+1}^N | x_n = a_i, Y_1^n\} \\ &= \Pr\{x_n = a_i, Y_1^n\} \cdot \Pr\{Y_{n+1}^N | x_n = a_i\} \end{aligned} \tag{3.7}$$

According to Eq. (3.7), the symbol distribution at time n given the past received symbols is statistically independent of the symbol distribution given future received symbols, within the observation window. Then, we define

$$\left.\begin{aligned} \alpha_n(a_i) &= \Pr\{x_n = a_i, Y_1^n\} \\ \beta_n(a_i) &= \Pr\{Y_{n+1}^N | x_n = a_i\} \end{aligned}\right\} \tag{3.8}$$

Thus,

$$\lambda_n(a_i) = \alpha_n(a_i) \cdot \beta_n(a_i) \tag{3.9}$$

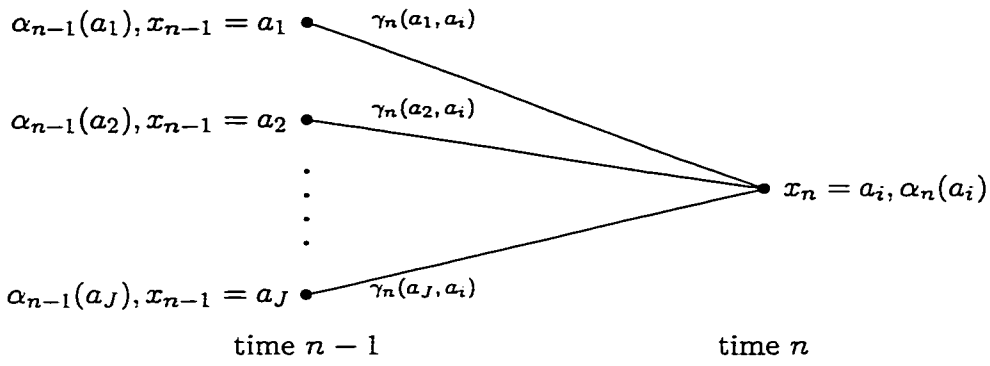

Figure 3.3: Computing the forward variable $\alpha_n(a_i)$

The forward variable $\alpha_n(a_i)$ is solved for inductively as follows

$$\begin{aligned}
\alpha_n(a_i) &= \sum_{j=1}^{J} \Pr\{x_{n-1} = a_j, x_n = a_i, Y_1^n\} \\
&= \sum_{j=1}^{J} \Pr\{x_{n-1} = a_j, x_n = a_i, Y_1^{n-1}, y_n\} \\
&= \sum_{j=1}^{J} \Pr\{x_{n-1} = a_j, Y_1^{(n-1)}\} \cdot \Pr\{x_n = a_i, y_n | x_{n-1} = a_j, Y_1^{(n-1)}\} \\
\alpha_n(a_i) &= \sum_{j=1}^{J} \Pr\{x_{n-1} = a_j, Y_1^{(n-1)}\} \cdot \Pr\{x_n = a_i, y_n | x_{n-1} = a_j\} \\
&= \sum_{j=1}^{J} \alpha_{n-1}(a_j) \cdot \gamma_n(a_j, a_i) \qquad n = 2, 3, ..., N \,\&\, i = 1, 2, \ldots, J
\end{aligned} \tag{3.10}$$

where

$$\gamma_n(a_j, a_i) = \Pr\{x_n = a_i, y_n | x_{n-1} = a_j\} \tag{3.11}$$

The variable $\gamma_n(a_j, a_i)$ represents the transition probability of hypothesis $x_{n-1} = a_j$ to the joint event of $x_n = a_i$ and received channel output y_n. Moreover, the variable $\alpha_{n-1}(a_j)$ is the joint probability that the sequence $\{y_1, y_2, \ldots, y_{n-1}\}$ is observed and the symbol at time $n-1$ is a_j. Therefore, the product $\alpha_{n-1}(a_j)\,\gamma_n(a_j, a_i)$ represents the probability of the joint event that the sequence $\{y_1, y_2, \ldots, y_n\}$ is observed and that the symbol a_i at time n is preceded by the symbol a_j at time $n-1$. The summation over all j, in Eq. (3.10), produces the joint event probability of observing

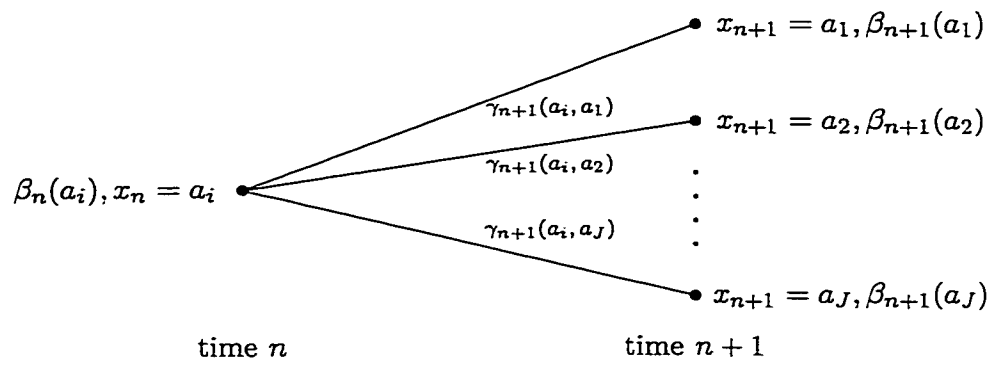

Figure 3.4: Computing the backward variable $\beta_n(a_i)$

Y_1^n and that the symbol at time n is a_i. Fig. 3.3 represents a graphical illustration of the way we compute the forward variable $\alpha_n(a_i)$.

Similarly, we can solve by induction for $\beta_n(a)$ in the following manner

$$\begin{aligned}
\beta_n(a_i) &= \sum_{j=1}^{J} \Pr\{x_{n+1} = a_j, Y_{n+1}^N | x_n = a_i\} \\
&= \sum_{j=1}^{J} \Pr\{x_{n+1} = a_j, y_{n+1}, Y_{n+2}^N | x_n = a_i\} \\
\beta_n(a_i) &= \sum_{j=1}^{J} \Pr\{x_{n+1} = a_j, y_{n+1} | x_n = a_i\} \cdot \Pr\{Y_{n+2}^N | x_n = a_i, x_{n+1} = a_j, y_{n+1}\} \\
&= \sum_{j=1}^{J} \Pr\{Y_{n+2}^N | x_{n+1} = a_j\} \cdot \Pr\{x_{n+1} = a_j, y_{n+1} | x_n = a_i\} \\
&= \sum_{j=1}^{J} \beta_{n+1}(a_j) \cdot \gamma_{n+1}(a_i, a_j) \quad n = N-1, \ldots, 2, 1 \,\&\, i = 1, 2, \ldots, J \quad (3.12)
\end{aligned}$$

The interpretation of Eq. (3.12) is illustrated in Fig. 3.4. Eq. (3.12) describes a backward recursion used to compute the probability of observing the sequence Y_{n+1}^N given that the symbol at time n is a_i. The backward procedure operates on the probability of the observation sequence after time $n+1$ given all possible symbols a_j (the variable $\beta_{n+1}(a_j)$), as well as the probability of observing y_{n+1} and to have had the symbol a_i (the variable $\gamma_{n+1}(a_i, a_j)$). Therefore, the forward-backward procedure computes $\alpha_n(a)$ in the forward direction ($n = 2, 3, \ldots, N$) at the same time we

Figure 3.5: Notation of the channel coding system.

receive y_n. However, the procedure evaluates $\beta_n(a)$ in the backward direction after receiving the sequence Y_1^N. The probability variable $\gamma_n(a_j, a_i)$ is computed according to

$$\begin{aligned} \gamma_n(a_j, a_i) &= \Pr\{x_n = a_i | x_{n-1} = a_j\} \cdot \Pr\{y_n | x_n = a_i, x_{n-1} = a_j\} \\ &= \Pr\{x_n = a_i | x_{n-1} = a_j\} \cdot \Pr\{y_n | x_n = a_i\} \end{aligned} \tag{3.13}$$

The second line of Eq. (3.13) follows the fact that the channel output y_n is completely defined by the channel input at time n.

Implementation Approach

1. Initialize $\alpha_1(a_i)$ and $\beta_N(a_i)$, $\forall\, a_i \in \mathcal{A}$.
2. Upon the reception of y_n the detector computes $\gamma_n(a_j, a_i)$ according to Eq. (3.13) and $\alpha_n(a_i)$ using Eq. (3.11). Store $\alpha_n(a_i), \gamma_n(a_j, a_i)$, $n = 1, 2, \ldots, N$, & $\forall\, a_i, a_j \in \mathcal{A}$.
3. At $n = N$, recursively compute $\beta_n(a_i)$ using Eq. (3.12), $n = N-1, \ldots, 2, 1$, $\forall\, a_i, a_j \in \mathcal{A}$. Then compute $\lambda_n(a_i)$ using Eq. (3.9), $\forall\, a_i \in \mathcal{A}$.
4. Then the set $\Pr\{x_n | Y_1^N\}; n = 1, 2, \ldots, N$ is computed by normalizing the set $\{\lambda_n(a_i)\}$, $\forall\, a_i \in \mathcal{A}$ to add up to one. Then, decide on $\hat{x}_n$.

Note that the values of $\alpha_n(a), \beta_n(a)$, $\forall\, a \in \mathcal{A}$ depend geometrically on past and future values of large number of terms. Therefore, it is very likely that their values get significantly large or vanishingly small. To solve this problem we scale $\alpha_n(a), \beta_n(a)$ at each n by $\sum_a \alpha_n(a)$, $\sum_a \beta_n(a)$, respectively, [96].

3.1.2 Symbol-MAP-Trellis-Decoding

In some applications the channel input sequence is produced by a finite state machine with a specific trellis structure. In this section we describe a simplified version of Bahl's algorithm, [6], which is presented in [92]. The algorithm follows a prespecified trellis structure to estimate the probability of an input to the finite-state

machine at time n, given a channel output observation sequence. In other words, according to a received observation sequence, the algorithm provides a probabilistic measure of the corresponding state sequence. The estimation approach is a forward-backward procedure similar to the one presented in the previous section. The following development is general for nonbinary-input finite-state machines.

In Fig. 3.5 we show a typical block diagram of a channel coding system. Let $w_n = i; i = 0, 1, \ldots, I-1$ represent the k-bit input symbol of the finite-state machine at time n, where $I = 2^k$. The channel encoder output is denoted by x_n and has a constraint length ν. At time n the encoder is in state $S_n = m;\ m = 0, 1, \ldots, M-1;\ M = 2^{k(\nu-1)}$. In order to perform MAP decoding, we decide on the symbol $\hat{w}_n$ with the maximum APP

$$\hat{w}_n = \arg \max_{i \in [0, I-1]} \Pr \{w_n = i | Y_1^N\} \tag{3.14}$$

where $Y_1^N = \{y_1, \ldots, y_n, \ldots, y_N\}$ is the channel output sequence. Now, if we define

$$\lambda_n^i(m) = \Pr \{w_n = i, S_n = m, Y_1^N\} \tag{3.15}$$

then, we compute the APP of a decoded data symbol as follows

$$\Pr \{w_n = i | Y_1^N\} = \frac{\sum_{m=0}^{M-1} \lambda_n^i(m)}{\sum_{m=0}^{M-1} \sum_{i=0}^{I-1} \lambda_n^i(m)} \tag{3.16}$$

For the case of memoryless channel and using Bayes' rule, the joint probability in Eq. (3.15) reduces to

$$\lambda_n^i(m) = \Pr \{w_n = i, S_n = m, Y_1^n\} \cdot \Pr \{Y_{n+1}^N | w_n = i, S_n = m\} \tag{3.17}$$

Eq. (3.17) states that at some point in the trellis the joint distribution of a state and a branch index given past received channel output symbols is statistically independent of the same distribution given future received symbols. Accordingly, we define the forward and backward variables in the following manner

$$\left.\begin{aligned} \alpha_n^i(m) &= \Pr \{w_n = i, S_n = m, Y_1^n\} \\ \beta_n^i(m) &= \Pr \{Y_{n+1}^N | w_n = i, S_n = m\} \end{aligned}\right\} \tag{3.18}$$

Then, it follows that

$$\lambda_n^i(m) = \alpha_n^i(m) \cdot \beta_n^i(m) \tag{3.19}$$

To compute the forward variable $\alpha_n^i(m)$, note that the following is true

$$\begin{aligned}
\alpha_n^i(m) &= \sum_{j=0}^{I-1}\sum_{\acute{m}=0}^{M-1} \Pr\{w_n = i, w_{n-1} = j, S_n = m, S_{n-1} = \acute{m}, Y_1^n\} \\
&= \sum_{j=0}^{I-1}\sum_{\acute{m}=0}^{M-1} \Pr\{w_{n-1} = j, S_{n-1} = \acute{m}, Y_1^{n-1}\} \\
&\quad \cdot \Pr\{w_n = i, S_n = m, y_n | w_{n-1} = j, S_{n-1} = \acute{m}\} \\
&= \sum_{j=0}^{I-1}\sum_{\acute{m}=0}^{M-1} \alpha_{n-1}^j(\acute{m}) \cdot \gamma_n^{j,i}(\acute{m}, m) \qquad (3.20)
\end{aligned}$$

where

$$\gamma_n^{j,i}(\acute{m}, m) = \Pr\{w_n = i, S_n = m, y_n | w_{n-1} = j, S_{n-1} = \acute{m}\} \qquad (3.21)$$

represents the joint probability of being in state m at time n and observing the output y_n following the branch i, given that we arrived at state $S_n = m$ after being in state $S_{n-1} = \acute{m}$ using the branch j. The variable $\gamma_n^{j,i}(\acute{m}, m)$, in Eq. (3.21) can be written as

$$\begin{aligned}
\gamma_n^{j,i}(\acute{m}, m) &= \Pr\{S_n = m | w_{n-1} = j, S_{n-1} = \acute{m}\} \\
&\quad \cdot \Pr\{w_n = i | S_n = m, w_{n-1} = j, S_{n-1} = \acute{m}\} \\
&\quad \cdot \Pr\{y_n | w_n = i, S_n = m, w_{n-1} = j, S_{n-1} = \acute{m}\} \qquad (3.22)
\end{aligned}$$

However, since we are following a trellis structure, there is only one state, denoted as $S_b^j(m)$, at which we may arrive if we go backward from state $S_n = m$ following the branch $w_{n-1} = j$. Consequently, the probability $\Pr\{S_n = m | w_{n-1} = j, S_{n-1} = \acute{m}\}$ has two possible values, 0 or 1. Furthermore, for uncorrelated sequence of symbols at the encoder input, the probability $\Pr\{w_n = i | S_n = m, w_{n-1} = j, S_{n-1} = \acute{m}\}$ simplifies to $\Pr\{w_n = i\}$. Also, a trellis path at time n is completely specified by the state S_n and the hypothesis w_n. Hence

$$\gamma_n^{j,i}(\acute{m}, m) = \gamma_n^i(m) = \Pr\{w_n = i\} \cdot \Pr\{y_n | w_n = i, S_n = m\} \qquad (3.23)$$

Based on the simplified version of $\gamma_n^i(m)$ given in Eq. (3.23), the forward variable $\alpha_n^i(m)$, defined in Eq. (3.20), is computed as follows

$$\alpha_n^i(m) = \gamma_n^i(m) \sum_{j=0}^{I-1} \alpha_{n-1}^j(S_b^j(m)) \quad n = 2, 3, \ldots, N \qquad (3.24)$$

A graphical representation of Eq. (3.24) is shown in Fig. 3.6.

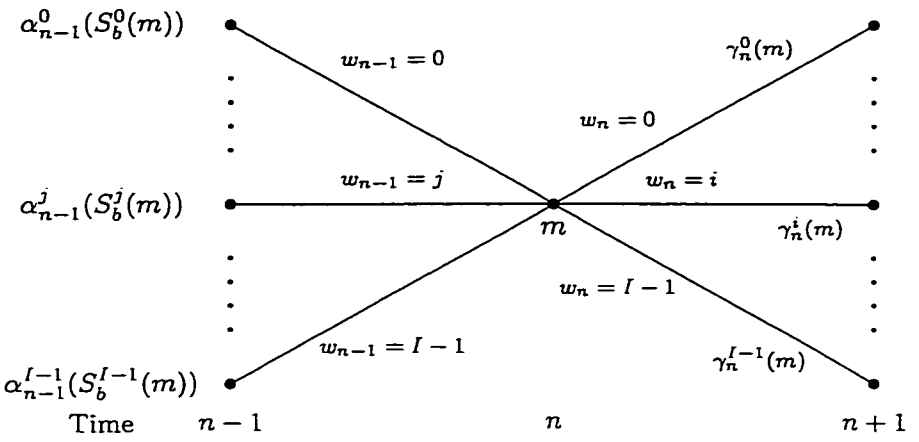

Figure 3.6: Computing the forward variable $\alpha_n^i(m)$

In order to inductively solve for the backward variable, note that

$$\beta_{n-1}^{j}(\acute{m}) = \Pr\left\{Y_n^N | w_{n-1}=j, S_{n-1}=\acute{m}\right\} \tag{3.25}$$

Then, we proceed as

$$\begin{aligned}\beta_{n-1}^{j}(\acute{m}) &= \sum_{i=0}^{I-1}\sum_{m=0}^{M-1} \Pr\left\{w_n=i, S_n=m, y_n, Y_{n+1}^N | w_{n-1}=j, S_{n-1}=\acute{m}\right\} \\ &= \sum_{i=0}^{I-1}\sum_{m=0}^{M-1} \Pr\left\{Y_{n+1}^N | w_n=i, S_n=m, y_n, w_{n-1}=j, S_{n-1}=\acute{m}\right\} \\ &\quad \cdot\Pr\left\{w_n=i, S_n=m, y_n | w_{n-1}=j, S_{n-1}=\acute{m}\right\} \end{aligned} \tag{3.26}$$

However, the trellis path information at time $n-1$ is irrelevant given the information at time n, and since the channel is memoryless, Eq. (3.26) simplifies to

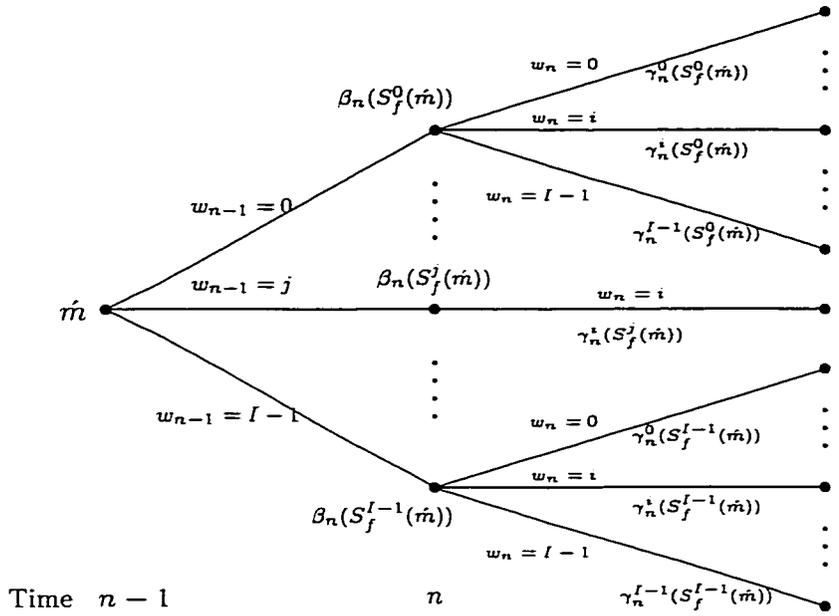

Figure 3.7: Computing the backward variable $\beta_{n-1}^j(\acute{m})$

$$\begin{aligned}\beta_{n-1}^j(\acute{m}) &= \sum_{i=0}^{I-1}\sum_{m=0}^{M-1} \Pr\{Y_{n+1}^N | w_n = i, S_n = m\} \\ &\quad \cdot \Pr\{w_n = i, S_n = m, y_n | w_{n-1} = j, S_{n-1} = \acute{m}\} \\ &= \sum_{i=0}^{I-1}\sum_{m=0}^{M-1} \beta_n^i(m)\gamma_n^{j,i}(\acute{m}, m) \end{aligned} \tag{3.27}$$

The trellis structure, once again, restricts the forward transition from state $S_{n-1} = \acute{m}$ along the branch $w_{n-1} = j$ to one specific state at time n, denoted by $S_f^j(\acute{m})$. Using Eq. (3.22) and Eq. (3.23) into Eq. (3.27) along with the consept of $S_f^j(m)$ produces

$$\beta_{n-1}^j(\acute{m}) = \sum_{i=0}^{I-1} \beta_n^i(S_f^j(\acute{m})) \cdot \gamma_n^i(S_f^j(\acute{m})) \quad n = N, N-1, \ldots, 2 \tag{3.28}$$

A graphical representation of Eq. (3.28) is shown in Fig. 3.7. Finally, the initialization of $\alpha_0^i(m)$ and $\beta_N^i(m)$ is application dependent. We will show later in this thesis the proper initialization for different applications.

Chapter 4

Soft Source Encoding

Several similarities exist between the areas of source compression and channel coding, which are mainly attributed to the duality between the rate distortion theory and the channel capacity dispute. Both source encoding and channel decoding are redundancy removal processes. Furthermore, both areas of research use the Euclidean space and employ lattice and trellis structures in the coding process. In chapter 2 we reviewed the concept of trellis vector quantization (TVQ). It has been shown that trellis waveform coding provide near theoretical limit distortion-rate performance [66], [9], [10]. The authors in [12] describe an efficient approach for TVQ system design using an LBG-based algorithm to search for an optimal minimum distortion partition. However, the LBG algorithm is extremely sensitive to initial codebooks, which control the descent to locally optimal configurations. Furthermore, this iterative optimization approach requires relatively long training sequences that are supposed to approximate the source statistical behavior, which is a suboptimal assumption in the case of nonstationary sources. Consequently, it is very likely that a nonconvex distortion surface tricks a poorly initialized LBG algorithm into settling at a locally optimal solution. In addition to these limitations, the codebook design problem of TVQ has another degree of freedom; that is the trellis size and the codeword/branch assignment issue. These limitations of the LBG-algorithm called for other global codebook search techniques, such as deterministic annealing [56], [97], [57], and simulated annealing [58], [24].

The heart of a trellis vector quantizer, which controls the encoding process, is the trellis search algorithm. As mentioned earlier, the Viterbi algorithm is one of the algorithms used to search for the minimum distortion sequence of symbols, by

choosing the corresponding minimum distortion trellis-path. The function of the Viterbi algorithm in searching for the minimum distortion path is analogous to its function in delivering the most probable transmitted sequence in channel decoding. In the context of channel decoding, it is possible to use either symbol-MAP or sequence-MAP algorithms, according to the application. Symbol-MAP decoding minimizes the probability of symbols in error, while sequence-MAP decoding minimizes the probability of sequences of symbols (words) in error. Nonetheless, for the special case of independent and identically distributed channel input symbols, the two algorithms are equivalent in the sense that both algorithms minimize the probability of sequence of symbols in error, since

$$\begin{aligned} \max\left\{\Pr\left\{X_1^N|Y_1^N\right\}\right\} &= \max\left\{\Pr\left\{x_1|Y_1^N\right\}\cdot\Pr\left\{x_2|Y_1^N,x_1\right\}\cdots\right. \\ &\quad \left.\Pr\left\{x_N|Y_1^N,x_{N-1},x_{N-2},\ldots\right\}\right\} \end{aligned} \tag{4.1}$$

Therefore, for independent x_n, Eq. (4.1) reduces to

$$\max\left\{\Pr\left\{X_1^N|Y_1^N\right\}\right\} = \max\left\{\prod_{n=1}^{N}\Pr\left\{x_n|Y_1^N\right\}\right\} \tag{4.2}$$

Clearly maximizing the term in Eq. (4.2) is equivalent to maximizing $\Pr\left\{x_n|Y_1^N\right\}$. The ultimate goal of any compression scheme, in particular TVQ, is to deliver a redundancy free output sequence. As explained in appendix B, the redundancy is due to the nonuniform distribution of the output symbols as well as the correlation between these symbols. Therefore, in anticipation of producing an uncorrelated and uniformly distributed sequence of symbols, a TVQ considers the branch indices as independent and identically distributed. Under this consideration, and given a source output sequence of vectors, using an algorithm that produces a sequence of minimum distortion symbols is equivalent to using the Viterbi algorithm, which delivers the minimum distortion sequence.

In this chapter, we introduce a new TVQ search algorithm. The new compression approach represents the "natural dual" to symbol-MAP trellis channel decoding. The search algorithm is based on the early forward-backward BCJR algorithm introduced for channel decoding in [6]. The algorithm performs decision-delayed processing and delivers for each time instant a set of reliability (soft) values that are used to decide on the minimum distortion source encoder output symbol. Since the algorithm delivers soft information, in the form of transition probabilities, we call this encoding approach "soft trellis vector quantization" (STVQ). It is worth mentioning that forward-backward algorithms are heavily used in finite-state Markov modeling and speech recognition systems, [98], [97], [99], [100], [101] and [102]. However, there is no compression associated with Markov modeling, the forward-backward

algorithm simply identifies the model parameters following a MAP estimation approach. The first part of this chapter addresses the derivation of the new soft compression algorithm. Besides that, we provide several simulation cases to show that this algorithm provides similar rate-distortion performance as compared to the performance of the Viterbi algorithm.

In section 4.3 of this chapter, we extend the new compression algorithm by using the reliability information to provide a robust fuzzy, yet deterministic, procedure for the design of trellis compression systems. The fuzzy design approach alleviates the problems associated with the LBG algorithm when applied to TVQ codebook design. The derivation of the new fuzzy trellis search algorithm is conceptually based on the development of the rate distortion theory, and is similar to deterministic annealing for data clustering, in particular vector quantization. We will show through several simulation examples that the new fuzzy design approach produces lower distortion configurations, while being significantly less sensitive to both the initial codebook and the length of the training sequence. Simulations involve the compression of mixture Gaussian and Gauss-Markov sources, as well as still images.

4.1 Source Encoding and Channel Decoding

In order to introduce the new compression algorithm in the next section, we first start by drawing the following analogy between channel decoding and source encoding. The two systems are presented in Fig. 4.1. The channel decoder operates on the channel output sequence Y_1^N to produce the maximum *a posteriori* probability sequence $\hat{X}_1^N$. This sequence represents the best estimate for the sequence X_1^N at the channel encoder input in the Bayesian sense. In the process of estimating $\hat{X}_1^N$, the channel decoder removes the controlled redundancy, which was introduced for error protection over the noisy channel. Following a similar approach, we use a source encoder to remove the redundancy in the information sequence and deliver the output sequence X_1^N. The symbols of the source encoder output sequence are ideally uncorrelated and uniformly distributed. Hypothetically speaking, the source encoder output sequence corresponds to the channel encoder input. Thus, we may consider the source of information as a communications channel, which corrupts the sequence X_1^N with un-wanted redundancy. However, unlike the channel decoding case, this redundancy is stochastic and uncontrolled.

We know that the measure of performance in TVQ coding systems is the average distortion of reproduction. Therefore, encoding is performed by choosing the

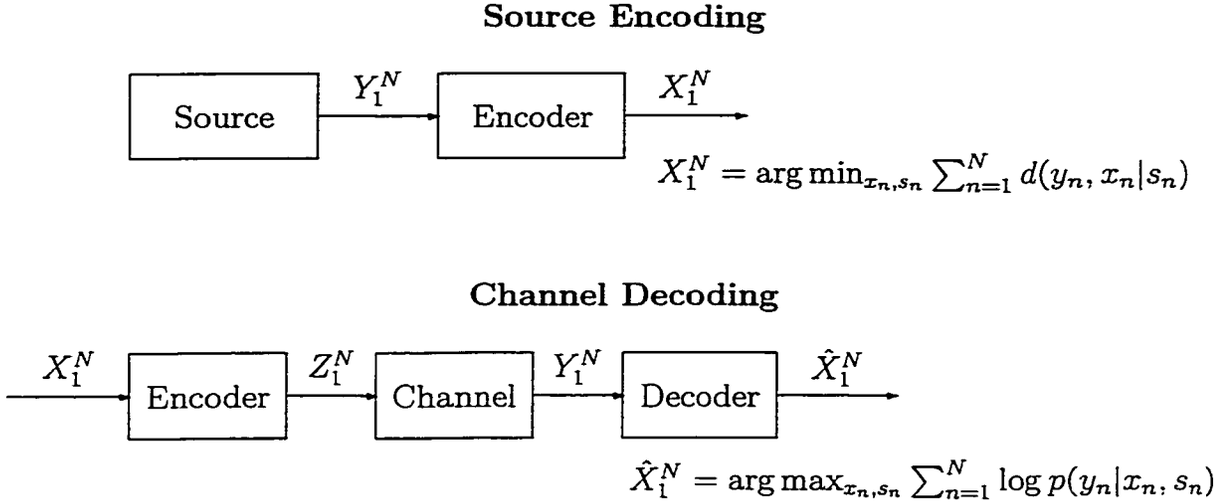

Figure 4.1: The analogy under consideration.

minimum distortion sequence according to

$$X_1^N = \arg \min_{X_1^N \in \mathcal{A}^N} D\{X_1^N|Y_1^N\} \tag{4.3}$$

where

$$D\{X_1^N|Y_1^N\} = \sum_{n=1}^{N} d(Y_n, x_n|s_n) \tag{4.4}$$

and $d(Y_n, x_n|s_n)$ is the Euclidean distance between the vector Y_n and the codeword associated with state $s_n \in \mathcal{S} = \{0, 1, \ldots, 2^{k(\nu-1)} - 1\}$ and branch index $x_n \in \mathcal{A} = \{0, 1, \ldots, 2^k - 1\}$. Accordingly, the Viterbi algorithm searches for the minimum distortion path by choosing, at each time instant n, the branch labeled by x_{n-T}, which leads to the minimum distortion path, where T represents the encoding depth. The search proceeds between a state s_{n-1} and another state s_n using the following metric

$$M_n^m = \min_{x_n \in \mathcal{A}, s_{n-1} \in \mathcal{S}} \{ M_{n-1}^{\acute{m}} + d(Y_n, x_n|s_n = m) \} \tag{4.5}$$

where $M_{n-1}^{\acute{m}}$ is the metric (cumulative distortion) at state $s_{n-1} = \acute{m}$, given that s_{n-1} is connected to s_n.

On the other hand, operating on a channel output sequence Y_1^N, a MAP channel decoder selects an estimate $\hat{Z}_1^N$ of the channel input sequence Z_1^N following the rule

$$\hat{Z}_1^N = \arg\max_{Z_1^N} \Pr\{Z_1^N | Y_1^N\} \tag{4.6}$$

Given that the channel input symbols $\{Z_n\}$, which color the branches of the decoding trellis, are independent and identically distributed, Eq. (4.6) reduces to the ML decoding rule

$$\hat{Z}_1^N = \arg\max_{Z_n} \prod_{n=1}^{N} \Pr\{Y_n | Z_n\} \tag{4.7}$$

We proceed by replacing the channel input hypothesis Z_n by the corresponding trellis state s_n and branch index x_n, as follows

$$\hat{X}_1^N = \arg\max_{x_n} \prod_{n=1}^{N} \Pr\{Y_n | x_n, s_n\} \tag{4.8}$$

Since the logarithm is a monotonic function and the probability values are always nonnegative, Eq. (4.8) is equivalent to

$$\hat{X}_1^N = \arg\max_{x_n} \sum_{n=1}^{N} \log p(Y_n | x_n, s_n) \tag{4.9}$$

where $p(Y_n|x_n, s_n)$ is substituted for $\Pr\{Y_n|x_n, s_n\}$ as the transition probability function between the channel output and the channel input, which is associated with x_n, s_n. It follows that, in order to implement the decision rule of Eq. (4.9), the Viterbi algorithm is used to follow the channel decoding trellis with the branch metric

$$M_n^m = \max_{x_n, s_{n-1}} \{ M_{n-1}^{\acute{m}} + \log p(Y_n | x_n, s_n = m) \} \tag{4.10}$$

where $M_{n-1}^{\acute{m}}$ and M_n^m are the metrics at states $s_{n-1} = \acute{m}$ and $s_n = m$, respectively.

The similarity between the two cases of employing the Viterbi algorithm for source encoding and channel decoding is mathematically supported by Eqs. (4.5) and (4.10). In fact, considering that the channel probability distribution is

$$p(Y_n | x_n, s_n) = \exp\{-d(Y_n, x_n | s_n)\} \tag{4.11}$$

then substituting this value of $p(Y_n|x_n, s_n)$ in Eq. (4.10), we get

$$M_n^m = \max_{x_n, s_{n-1}} \{ M_{n-1}^{\acute{m}} - d(Y_n, x_n | s_n = m) \} \tag{4.12}$$

Note that the metric in Eq. (4.12) is identical to the metric in Eq. (4.5). Hence, the problem of trellis vector quantization is equivalent to the decoding problem of a convolutionally encoded sequence transmitted over a channel characterized by Eq. (4.11). Therefore, it is correct in principle to perform the compression process using the exponential distribution given by Eq. (4.11). This distribution is used in the next section with the forward-backward algorithm to develop a new soft trellis compression algorithm.

In the context of data clustering, the conditional pdf given by Eq. (4.11) represents an association probability that describes the degree of membership of the vector Y_n to the set represented by the codeword addressed by x_n, s_n. This distribution is a member of a family of distributions known as the Gibbs distribution, which represents the solution to the optimization problem in fuzzy clustering [57], as described in chapter 2. The Gibbs distribution is controlled by a Lagrange multiplier η as

$$p(Y_n|x_n, s_n) = \frac{\exp\{-\eta d(Y_n, x_n|s_n)\}}{\mathcal{Z}} \tag{4.13}$$

where $\mathcal{Z}$ is a normalizing factor. It is clear that this distribution assigns higher probability values to lower energy configurations. Furthermore, for vanishingly small values of η, the distribution is uniform in the sense that it equally likely assigns the vector Y_n to the different clustering sets. However, as we increase η, higher clustering probabilities are expected with the sets that correspond to lower distortions. Therefore, rather than adopting the special case of $\eta = 1$, suggested by Eq. (4.11), we shall use this degree of freedom (the choice of η) to optimize the performance of the new compression algorithm for different sources and compression rates.

4.2 Soft Trellis Vector Quantization

In this section we use the analogy developed thus far, between the source and the channel, to introduce a new TVQ search algorithm [103] and [104]. The algorithm is based on the fact that we may consider the TVQ process as decoding at the output of the channel defined by Eq. (4.11). In chapter 3 we introduced a symbol-MAP algorithm, which uses a forward-backward procedure to find the *a posteriori* probability of a transmitted symbol given a channel output sequence of symbols. In a similar approach we use the algorithm presented in section 3.1.2 to develop a TVQ search algorithm to deliver a sequence of minimum distortion symbols.

Consider that the output vector of a stationary discrete-time source at time n is denoted by $Y_n = [y_{n1}, y_{n2}, \ldots, y_{nL}]$. The L-dimensional TVQ is characterized by a finite state decoder of k-bit input and constraint length ν, consequently, a trellis of $M = 2^{k(\nu-1)}$ states. A TVQ system uses an N-length source output sequence $Y_1^N = \{Y_1, Y_2, \ldots, Y_N\}$ and delivers for each time instant a branch index $x_n \in \mathcal{A} = \{0, 1, \ldots, I-1\}$, where $I = 2^k$. In order to produce a sequence of minimum distortion symbols, we first assume that the source is the channel described by Eq. (4.11). Then, given a source output sequence Y_1^N, we choose an output symbol according to

$$x_n = \arg\max_{i \in \mathcal{A}} \Pr\{x_n = i | Y_1^N\} \tag{4.14}$$

Now we follow the same approach presented in section 3.1.2 to solve for the set of probabilities $\Pr\{x_n = i|Y_1^N\}$, $i = 0, 1, \ldots, I-1$, at each time instant n. We proceed by defining the joint probability variable

$$\lambda_n^i(m) = \Pr\{x_n = i, s_n = m, Y_1^N\} \tag{4.15}$$

where $s_n = m$ is the encoder state at time n. It follows that

$$\Pr\{x_n = i|Y_1^N\} = \frac{\sum_{m=0}^{M-1} \lambda_n^i(m)}{\sum_{m=0}^{M-1}\sum_{i=0}^{I-1} \lambda_n^i(m)} \tag{4.16}$$

Recall that the problem of TVQ with the Viterbi algorithm is similar to Viterbi decoding at the output of a memoryless channel. In other words, trellis waveform coding is performed without considering the statistical correlation between successive source output vectors. In fact, appendix D shows the irrelevance of considering the source correlation to the trellis search process. Therefore, based on the same analogy the source is considered memoryless in the derivation of the new algorithm. Accordingly, we write Eq. (4.15) in the form

$$\lambda_n^i(m) = \alpha_n^i(m) \cdot \beta_n^i(m) \tag{4.17}$$

where

$$\left.\begin{aligned} \alpha_n^i(m) &= \Pr\{x_n = i, s_n = m, Y_1^n\} \\ \beta_n^i(m) &= \Pr\{Y_{n+1}^N | x_n = i, s_n = m\} \end{aligned}\right\} \tag{4.18}$$

Similar to the development of section 3.1.2, we can show that the forward variable is evaluated using

$$\alpha_n^i(m) = \gamma_n^i(m) \sum_{j=0}^{I-1} \alpha_{n-1}^j(S_b^j(m)) \quad n = 2, 3, \ldots, N \tag{4.19}$$

where $S_b^j(m)$ is the state at which we arrive if we go backwards from state $s_n = m$ along the branch $x_{n-1} = j$. Besides that,

$$\gamma_n^i(m) = \Pr\{Y_n | x_n = i, s_n = m\} \tag{4.20}$$

Note that in deriving Eq. (4.20), the symbols $\{x_n\}_{n=1,\ldots,N}$ are considered uncorrelated and identically distributed . This assumption is a direct consequence of the ultimate goal of producing a redundancy free output sequence. Moreover, this condition forces the search algorithm to equally test the different codewords that color the trellis branches, which is a basic procedure in trellis waveform coding. Then, Eq. (4.13) is used without the normalizing factor $\mathcal{Z}$ to substitute for the channel transition probability in Eq. (4.20), as follows

$$\gamma_n^i(m) = \exp\{-\eta d(Y_n, x_n = i | s_n = m)\} \tag{4.21}$$

The normalization by the factor $\mathcal{Z}$ is substituted by another normalizing approach, which will guarantee the stability of the algorithm as explained in the next section. On the other hand, the backward variable is computed as in section 3.1.2 using

$$\beta_{n-1}^j(\acute{m}) = \sum_{i=0}^{I-1} \beta_n^i(S_f^j(\acute{m})) \cdot \gamma_n^i(S_f^j(\acute{m})) \quad n = N, N-1, \ldots, 2 \tag{4.22}$$

where $S_f^j(\acute{m})$ is the state at which we arrive if we go forward from state $s_{n-1} = \acute{m}$ along the branch $x_{n-1} = j$.

4.2.1 Implementation Approach

One critical point to consider is the initialization of $\alpha_1^i(m)$ and $\beta_N(m)$. In this particular application, the processing of a block of source vectors starts from the zero state, thus, it is natural to initialize the forward variable as

$$\alpha_1^i(m) = \begin{cases} \gamma_1^i(0) & m = 0 \\ 0 & m \neq 0 \end{cases} \quad \forall i \tag{4.23}$$

Then, we proceed in the computation of $\alpha_n^i(m)$ for $n = 2, 3, \ldots, N$ according to Eq. (4.19). On the other hand, initializing $\beta_N^i(m)$ depends on whether we know the final state or not. Practically, we can choose to start encoding from a specific state, however, we cannot choose the last state that is controlled by the source output sequence. Therefore, the final states should have the same significance, which leads to the following initialization of the backward variable

$$\beta_N^i(m) = \frac{1}{M}, \quad \forall m \tag{4.24}$$

Then, Eq. (4.22) is used to compute $\beta_{n-1}^i(m)$ for $n = N, N-1, \ldots, 2$. After evaluating $\alpha_n^i(m)$ and $\beta_n^i(m)$, we compute $\lambda_n^i(m)$ using Eq. (4.17). Then, we evaluate the set of probabilities of interest using Eq. (4.16). Encoding is performed by delivering the $x_n = i$ associated with the maximum probability value, as indicated by Eq. (4.14).

It is clear that the values of the forward and backward variables depend geometrically on past and future values of large number of terms. Consequently, it is very likely that their values get significantly large or small. To solve this problem we scale each $\alpha_n^i(m)$ and $\beta_n^i(m)$ by $\sum_i \sum_m \alpha_n^i(m)$ and $\sum_i \sum_m \beta_n^i(m)$, respectively [96]. This normalization step compensates for the fact that we are not using the variable $\mathcal{Z}$ shown in Eq. (4.13). In the simulations section, we will demonstrate the significance of the proper choice of η in optimizing the performance of the algorithm in different applications. The procedure of this soft trellis vector quantization algorithm (STVQ) is shown in Table 4.1. In this table C_i^m is used to denote the reproduction codeword that colors the ith branch out of state m.

Table 4.1: The STVQ algorithm: sequence of implementation.

Initialize:

For $i = 0$ to $I-1$, $m = 0$ to $M-1$,

$$d(i,m) = [Y_1 - C_i^m]^2, \qquad \gamma_1^i(m) = \exp\{-\eta d(i,m)\}.$$

For $i = 0$ to $I-1$,

$$\alpha_1^i(0) = \gamma_1^i(0),\ \alpha_1^i(m) = 0;\ \forall m \neq 0.\ \beta_N^i(m) = \frac{1}{M}$$

Forward:

For $n = 2$ to N,

For $i = 0$ to $I-1$, $m = 0$ to $M-1$,

$$d(i,m) = [Y_n - C_i^m]^2, \qquad \gamma_n^i(m) = \exp\{-\eta d(i,m)\}$$

$$\alpha_n^i(m) = \gamma_n^i(m) \cdot \sum_{j=0}^{I-1} \alpha_{n-1}^j(S_b^j(m))$$

Normalize $\alpha_n^i(m)$ by $\sum_j \sum_m \alpha_n^j(m)$; $\forall i, m$.

Backward:

For $n = N$ to 2,

For $j = 0$ to $I-1$, $\acute{m} = 0$ to $M-1$,

$$\beta_{n-1}^j(\acute{m}) = \sum_{i=0}^{I-1} \beta_n^i(S_f^j(\acute{m})) \cdot \gamma_n^i(S_f^j(\acute{m}))$$

Normalize $\beta_{n-1}^j(\acute{m})$ by $\sum_j \sum_{\acute{m}} \beta_{n-1}^j(\acute{m})$; $\forall j, \acute{m}$.

Compute Probabilities:

For $n = 1$ to N, $i = 0$ to $I-1$,

$$\Pr\{x_n = i | Y_1^N\} = \frac{\sum_m [\alpha_n^i(m) \cdot \beta_n^i(m)]}{\sum_m \sum_j [\alpha_n^j(m) \cdot \beta_n^j(m)]}$$

Decide on the most probable x_n.

4.2.2 Computational Complexity

The forward-backward MAP algorithm, which is presented in chapter 3 and used in this chapter for compression, is capable of perfectly estimating the *a posteriori* probability of an information symbol. The attractive feature of this algorithm is that it provides soft reliability values for the different information symbols. Nevertheless, the algorithm is computationally complex in the sense that it performs multiplication, in the forward and backward directions, for different values of probabilities and nonlinear functions. In order to solve the complexity problem of this algorithm, several publications considered performing the entire procedure in the log-domain [92], [105], [38] and [106]. In the log-domain, multiplications are transformed into additions, which significantly reduces the complexity issue. The transformation into the log-domain resulted in two MAP algorithms, one optimal and is called the log-MAP algorithm, while the other is suboptimal and is denoted by the max-log-MAP algorithm.

The performance of the optimal log-MAP algorithm is identical to that of the conventional MAP algorithm. The derivation of this algorithm is based on the following Jacobian logarithm

$$\log(e^a + e^b) = \max(a, b) + \log(1 + e^{-|a-b|}) \tag{4.25}$$

Therefore, additions of exponentials performed in the log-domain are equivalent to choosing the maximum with a correction value given by $\log(1 + e^{-|a-b|})$. To further reduce the complexity of the log-MAP algorithm, certain range of the correction function is stored in a look-up table. The authors in [105] show that it is enough to store 8 values of the correction function to provide a near optimal performance. The transformation of the symbol-MAP algorithm to the equivalent log-MAP algorithm is performed by using the rule of Eq. (4.25) in solving for the logarithms of the forward and backward variables. On the other hand, the max-log-MAP algorithm drops the correction function $\log(1 + e^{-|a-b|})$ in Eq. (4.25) to reduce the complexity, however, with a suboptimal performance. It is shown in [106] and [105] that the log-MAP algorithm is about 4 times more complex than the Viterbi algorithm, which is approximately 2 times less complex than the max-log-MAP algorithm.

4.3 Performance Evaluation of STVQ

We provide in this section the rate-distortion simulation results of encoding a memoryless Gaussian source, and a Gauss-Markov source. The performance results are measured in terms of signal to MSE quantization noise ratios (SNR), which is defined in Eq. (C.1) in appendix C. A training sequence of 30,000 samples is used to search for a locally optimal codebook using the LBG-algorithm [12]. Longer training sequences usually provide better approximation of the source distribution. However, the adopted training sequence length provided the computational distortion, which is expected with TVQ for Gaussian sources [12], [51]. The other critical issue with the TVQ codebook design is the initial trellis codebook. Simulation results showed the significance of the initialization techniques developed in [51]. For the memoryless Gaussian source with compression rate 1 bit/sample (bps), we use a random initial codebook drawn from a zero-mean and 0.75 variance Gaussian distribution. This initialization is mathematically supported by the rate distortion theory [2]. In [2] Eq. (4.3.35), Berger shows that the distribution of the optimal reproduction alphabets for encoding a σ_y^2-variance Gaussian source is Gaussian with variance $\sigma_y^2 - D(R)$, where D(R) is the distortion rate value of the source. On the other hand, initializing the trellis branches with $\pm\sigma_y$ proved to be effective for the case of the Gauss-Markov source, where σ_y^2 is the source variance. Furthermore, experimental results showed that the performance of the STVQ algorithm improves as we increase the encoding depth, however, no significant improvement is achieved after an encoding depth of approximately $130k\nu$ vectors. On the other hand, the survivor paths using the Viterbi algorithm seem to converge after processing $40k\nu$ vectors.

In order to show the average performance of the two compression approaches, the optimal codebook is employed to find the average SNR value as well as the corresponding 95 percent confidence interval (CI). For this purpose, we use 100 sample sequences drawn using different random seeds from the same Gaussian distribution, each of which has 1000 samples. The computation of the confidence interval is based on the fact that the computed distortion for each sample sequence is a sample average. Thus, we can apply the central limit theorem to claim that the distribution of the computed sample distortions is Gaussian. Let the mean and the variance of the computed sample distortions be $\mathcal{D}_{av}$ and σ_{av}^2, respectively. It follows that there is a 95% chance that an observation (computed distortion) will fall within $\mathcal{D}_{av} \pm 1.96\sigma_{av}^2$. Then, $\mathcal{D}_{av}$ is used to compute the average SNR in dB, and the $\pm 1.96\sigma_{av}^2$ is used to compute the corresponding confidence interval in dB. Although the confidence interval in dB is not symmetric around the average SNR, for simplicity of presentation we shall use symmetric CI considering the larger off-average values.

Table 4.2: The VTVQ and STVQ 1 bps coding performance for the memoryless Gaussian source.

	STVQ		VTVQ	
ν	SNR (dB)	CI (dB)	SNR (dB)	CI (dB)
2	4.747	±0.009	4.734	±0.009
3	4.843	±0.007	4.848	±0.007
4	5.140	±0.004	5.174	±0.004
5	5.147	±0.003	5.150	±0.006
6	5.283	±0.004	5.233	±0.006

4.3.1 Memoryless Gaussian Source

In the first case study we consider the $R = k/L = 1/1 = 1$ bps encoding of a memoryless unit-variance Gaussian source. As shown in appendix A, the MSE-distortion rate function of a Gaussian source is given by [2]

$$D_G(R) = 2^{-2R}\sigma_y^2 \tag{4.26}$$

where R is the compression rate in bps. Thus, for a unit-variance source and rate 1 bps, $D_G(1) = 0.25$, which corresponds to an SNR= 6.02 dB. In Table 4.2, we show the simulation results for several trellis constraint lengths ν, where we observe the very close performance of the two algorithms.

4.3.2 First-order Gauss-Markov Source

The close performance of the STVQ and the VTVQ approaches is also demonstrated through encoding a first-order Gauss-Markov source with correlation coefficient $\rho = 0.9$. A brief introduction to sources with memory, and the AR(1) source in particular, is presented in appendix A. The MSE-distortion rate function for a first order Gauss-Markov source is [2]

$$D_G(R) = 2^{-2R}(1-\rho^2)\sigma_y^2 \tag{4.27}$$

The validity of this equation is constrained by Eq. (A.18) of appendix A. In the first case, encoding at a rate $R = k/L = 1/1$ bps is performed for several values of

Table 4.3: The VTVQ and STVQ 1-bit coding performance for the AR(1) source, $\rho = 0.9$.

	STVQ		VTVQ	
ν	SNR (dB)	CI (dB)	SNR (dB)	CI (dB)
2	6.784	±0.195	6.790	±0.204
3	8.603	±0.146	8.629	±0.145
4	9.915	±0.072	9.950	±0.069
5	10.629	±0.038	10.682	±0.037

ν. According to Eq. (4.27) the highest theoretical SNR for this source at a rate of 1 bps is 13.23 dB. The computational rate 1 distortion results for this source using the two algorithms are shown in Table 4.3. To observe the effect of varying the compression rate on the reconstructed SNR, the performance of the two algorithms at the rate $R = 1/2$ bps is depicted in Table 4.4. Using the results presented in Fig. A.1 of appendix A, the theoretically attainable SNR at rate 1/2 for this source is 9.867 dB.

4.3.3 Discussion: Choosing η

Thus far, we have applied the forward-backward symbol-MAP algorithm for trellis waveform compression. The new algorithm delivers a sequence of minimum distortion symbols, which performs in the proximity of the minimum distortion sequence generated by the Viterbi algorithm. Simulation examples showed that the behavior of the STVQ algorithm is closely related to the value of η in Eq. (4.21). This was expected, since the value of η controls the degree of association to the different encoding sets that are represented by the reproduction codewords. Nonetheless, experimental results showed that there is an upper optimal value η^*, in the vicinity of which the STVQ algorithm provides its best performance. Note that the STVQ algorithm computes the forward and backward variables using the basic probability function defined in Eq. (4.21). Thus, for $\eta >> \eta^*$, some of the probability values computed using Eq. (4.21) become vanishingly small. As a result, this situation drives either all the forward variables or/and the backward variables to zero at some intermediate time instants. Thereby, limiting the performance of the STVQ algorithm by delivering misleading reliability information.

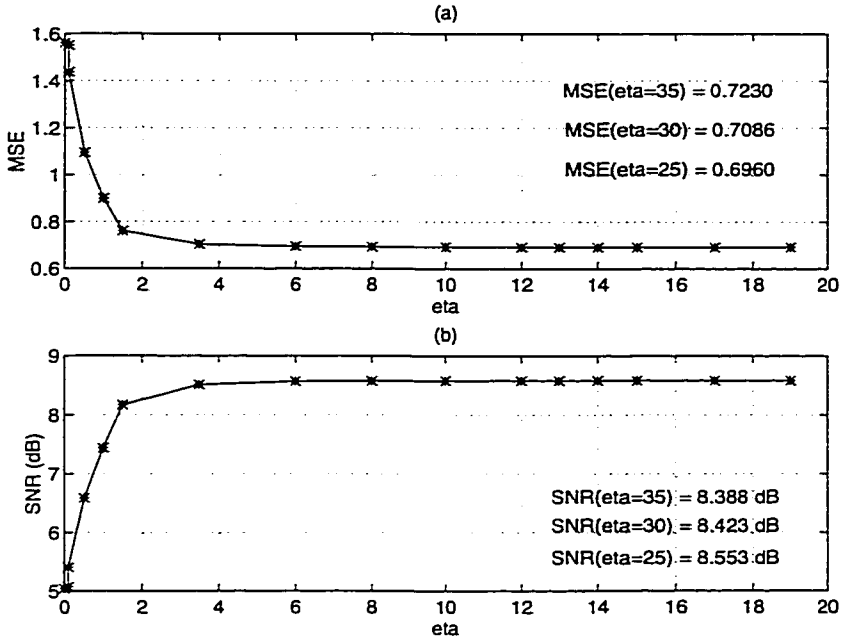

Figure 4.2: The performance of the STVQ algorithm as a function of η for encoding the AR(1) source at $R = 1$ pbs and $\nu = 3$: (a) MSE, (b) SNR in dB.

To further explain the malfunction of the STVQ algorithm at $\eta >> \eta^*$, the STVQ algorithm is programmed using the normalizing factor $\mathcal{Z}$ defined in Eq. (4.13). At high η, note that association is performed almost as hard clustering; i. e., the transition probability $\gamma_n^i(m)$ defined in Eq. (4.21) and normalized by $\mathcal{Z}$ will have an effective value of 1 for the branch corresponding to the minimum distortion codeword at time n. At the same time, for the other branches that practically failed the nearest neighbor test, $\gamma_n^i(m) = 0$. This consequently causes the computation along the trellis to disconnect, which is reflected in delivering zero forward and backward variables. Although hard clustering is the correct approach for VQ, the goal in TVQ is to deliver the minimum distortion sequence, which is performed by STVQ, rather than delivering the minimum distortion symbol at each time instant without considering the rest of the options for the entire encoding window. On the other hand, since the exponential term in Eq. (4.21) is controlled by both η and the instantaneous distortion, we should expect to use smaller values of η^* with larger average distortions of reproduction. In fact experimental results showed that η^* is proportional to $\nu/(DL)$.

As an example we apply the STVQ algorithm to search for the optimal codebook for the AR(1) source used earlier, with rate 1 bps and $\nu = 3$. The average distortions

Table 4.4: The VTVQ and STVQ 1/2-bit coding performance for the AR(1) source, $\rho = 0.9$.

	STVQ		VTVQ	
ν	SNR (dB)	CI (dB)	SNR (dB)	CI (dB)
2	6.030	±0.110	5.973	±0.107
3	7.377	±0.071	7.431	±0.088
4	8.190	±0.045	8.204	±0.033
5	8.553	±0.042	8.439	±0.018

achieved using several values of η are shown in Fig. 4.2. In an attempt to extend the performance of the STVQ algorithm for values of $\eta > \eta^*$, we uniformly re-initialized the forward and the backward variables whenever the sum of their values goes to zero at any time instant within the encoding window. For this particular example, $\eta^* \approx 13$, after this value the STVQ algorithm starts delivering zero sums of the forward and the backward variables. Nevertheless, as shown in Fig. 4.2 we succeeded to put the trellis computation back on track following the suggested re-initialization procedure. However, for extreme values of η, the zero sums start occurring more frequently, which naturally reduces the reliability of the compression algorithm.

The preceding discussion sets a practical procedure that we can follow to find η^*. Simply we start by apply the STVQ algorithm using relatively high values of η, these values are then reduced until no zero sums of forward or backward variables are detected. At this point, we can use that η as the optimal value of operation η^*. Note that relatively small values of η produce higher expected distortions of reproduction, as shown in Fig. 4.2 for $\eta < 3$. Further discussion of the significance of η as well as its relation to the rate distortion theory is provided in the development of section 4.4.

4.4 Fuzzy Soft Trellis Vector Quantization

The challenge of designing an efficient trellis waveform compression system starts with the design of the reproduction codebook. The LBG algorithm is a widely used codebook search algorithm with trellis coding systems [12]. We mentioned at the beginning of this chapter that the performance of the LBG algorithm is very sensitive to initial codebooks as well as to the lengths of the training sequences. Consequently, it is very likely that a nonconvex distortion surface tricks a poorly initialized LBG algorithm into settling at a locally optimal solution. A rather successful approach that was followed to alleviate this problem in the LBG algorithm is the method of simulated annealing (SA) [107], [108]. We stated in section 2.2 the disadvantages of this random search process along with several references that consider SA for source coding. Furthermore, we heuristically introduced in section 2.2 the non-random alternative to SA; that is deterministic annealing (DA). We showed how the DA process functions in a fuzzy clustering approach in an attempt to descend to a globally optimal configuration. In the sequel we shall use the development of the rate distortion theory to extend the STVQ algorithm to provide an efficient and robust fuzzy and deterministic trellis codebook search approach [104]. We will demonstrate in this section that the developed approach is capable of descending to lower distortion points while being highly insensitive to the initialization of the search process as well as to the length of the training sequence.

We shall start our development by visiting a basic theorem in the rate distortion theory, which was used to derive the famous Blahut algorithm. This theorem is then used to explain the computational rate distortion behavior of the STVQ algorithm in the process of deriving the new fuzzy compression approach. In [44], theorem 6.3.3 in particular, Blahut parameterizes the rate distortion function $R(D)$ in an equation, rather than in an inequality, using a Lagrange multiplier κ. Blahut manages to perform this parameterization after proving that $R(D)$ is a decreasing, continuous and convex function of D in the interval $0 \leq D \leq D_{max}$. In that theorem, the rate distortion function is expressed as [44]

$$R(D) = \kappa D + \min_{q} \min_{Q} \left[\sum_{j} \sum_{l} p_j Q_{l|j} \log \frac{Q_{l|j}}{q_l} - \kappa \sum_{j} \sum_{l} p_j Q_{l|j} d_{jl} \right] \tag{4.28}$$

where

$$D = \sum_{j} \sum_{i} p_j Q^{*}_{l|j} d_{jl} \tag{4.29}$$

where j and l are used to index the input and reproduction alphabets, respectively. $Q_{l|j}$ is the conditional probability function between l and j, and $Q^{*}_{l|j}$ is the distribution that satisfies the minimum in Eq. (4.28). Besides that, d_{jl} is the distortion

between symbols l and j, p_j is the input distribution, while q_l is the output distribution. The optimal conditional distribution that achieves the minimum value of $R(D)$ is shown to be [44]

$$Q_{l|j} = \frac{q_l \, e^{\kappa d_{jl}}}{\sum_l q_l \, e^{\kappa d_{jl}}} \tag{4.30}$$

This theorem is significant in the sense that it culminated into the famous Blahut algorithm, which computes certain number of points on the rate distortion curve parameterized by the corresponding values of the Lagrange multiplier κ. For a particular value of $\kappa \in]-\infty, 0]$, Blahut algorithm iteratively solves for the best distribution of the reproduction alphabets q_l, which along with $Q_{l|j}$ defined in Eq. (4.30) achieves a point on the rate distortion curve. In other words, Blahut algorithm characterizes the distribution of the optimal reproduction alphabets (q_l) as well as the transition probabilities ($Q_{l|j}$), which control the encoding rule, for a given value of κ.

At this stage we are interested in incorporating the rate distortion formulation into the derivation of an efficient fuzzy soft trellis vector quantization (FSTVQ) algorithm. We demonstrated in the previous section that the derivation of the STVQ algorithm is based on the association distribution given in Eq. (4.21) and parameterized by the Lagrange multiplier η. This distribution is clearly identical to the conditional distribution $Q_{l|j}$ shown in Eq. (4.30), which leads to an optimal rate-distortion point parameterized by κ. The two distributions are different in their consideration of the distribution of the output alphabet (q_l). While it is the goal to search for an optimal output alphabet distribution (q_l) in the development of the rate-distortion theory, practical compression systems, and the STVQ algorithm in particular, adopt the uniform distribution to describe the reproduction alphabets. This consideration reduces the degrees of freedom in practical systems, and allows for the exact computation of the output alphabets.

The other point of interest is the similarity between the parameters η and κ, where η in Eq. (4.21) is equivalent to $-\kappa$ in Eq. (4.30). It is more perceptive to interpret the independent Lagrange multiplier κ as the slope of the tangent at the corresponding point on the rate distortion curve. Accordingly, as $\kappa \rightarrow -\infty$ we approach the low-distortion high-rate points of the $R(D)$ curve, while $\kappa \rightarrow 0$ corresponds to the flat high-distortion low-rate points of the same curve. On the other hand, we showed in the previous section, in Fig. 4.2, that the performance of the STVQ algorithm is closely related to the choice of η. Unlike Blahut algorithm, which functions to find a rate-distortion point given the independent parameter κ, the TVQ problem involves encoding given a prespecified compression rate value. Therefore, the critical issue with the STVQ algorithm is to find the optimal η given a

compression rate and an anticipated average distortion value. Experimental results showed that we can expected the values of η to control the computational rate-distortion curve just like the values of κ control the theoretical rate-distortion curve. In other words, these results showed that the value of the "optimal" $\eta \in [0, \infty[$ is inversely proportional to the expected average distortion value.

To further show the relationship between the rate distortion formulation and practical source coding systems, we consider the general problem of data clustering. Irrespective of whether clustering is regular or performed over a trellis, the system design criterion is to minimize the average distortion

$$D = \sum_j \sum_l p_j \pi_{l|j} d_{jl} \tag{4.31}$$

since we are practically interested in delivering a hard decision, the conditional distribution $\pi_{l|j}$ is defined as

$$\pi_{l|j} = \begin{cases} 1, & j \in S_l \\ 0, & \text{Otherwise} \end{cases} \tag{4.32}$$

where S_l represents the lth encoding set. Furthermore, since we do not usually have access to the source distribution p_j, stochastic averaging in Eq. (4.31) is replaced by time averaging. Unlike the distortion measure defined in Eq. (4.29), which includes the soft conditional encoding rule ($Q_{l|j}$), the practical measure of Eq. (4.31) is based on a hard association rule ($\pi_{l|j}$).

In order to fit the notion of STVQ, we redefine the distribution $\pi_{l|j}$ as follows

$$\pi_n(l|Y_1^N) = \begin{cases} 1, & \psi_n(l) > \psi_n(j) \\ 0, & \text{otherwise} \end{cases} \quad ; \quad l, j = 0, 1, \ldots, 2^{k\nu} - 1 \tag{4.33}$$

where $\psi_n(l) = \Pr\{C_l|Y_1^N\}$, and C_l is a codeword associated with a trellis state $s_n = m$ and a branch $x_n = i$. Accordingly,

$$\psi_n(l) = \frac{\lambda_n^i(m)}{\sum_{m=0}^{M-1} \sum_{i=0}^{I-1} \lambda_n^i(m)} \tag{4.34}$$

with $\lambda_n^i(m)$ defined in Eq. (4.15). We mentioned earlier that for a particular compression rate and expected average distortion there is a proper value η^* in Eq. (4.21), below which the STVQ-codebook search process with the LBG algorithm converges to higher average distortion values. Therefore, considering that we are descending on the computational distortion rate curve from high distortion values (small η) to the distortion corresponding to the operational rate (at η^*), we can perform some

sort of deterministic relaxation by minimizing a soft average distortion while incrementing the value of η from small values to η^*. This intuitively suggests using the soft information $\psi_n(l)$, which is delivered by the STVQ algorithm and defined in Eq. (4.34), in place of the hard association rule $\pi_n(l|Y_1^N)$ of Eq. (4.33). Accordingly, instead of minimizing the average distortion given by Eq. (4.31) with $\pi_{l|j}$ defined in Eq. (4.33), we propose minimizing the following distortion measure in the process of searching for the optimal codebook

$$D = \frac{1}{N_T} \sum_n \sum_{l=0}^{2^{k\nu}-1} \psi_n(l)\, d(Y_n, C_l) \tag{4.35}$$

where N_T is the length of the training sequence. It follows that to minimize the distortion given by Eq. (4.35) we set the gradient of D with respect to $\{C_l\}$ to zero, which for the squared error distortion produces the centroid update equations

$$C_l = \frac{\sum_n \psi_n(l) Y_n}{\sum_n \psi_n(l)} \qquad \forall\, C_l \tag{4.36}$$

According to Eq. (4.36), a source output vector Y_n contributes a $\psi_n(j)$-weight of its value to the update of each reproduction codeword C_j. Clearly this update equation provides the deterministic relaxation sought for to help the LBG algorithm finding the global minimum configuration. Eq. (4.36) with the STVQ algorithm defines a new fuzzy search process we refer to as fuzzy-STVQ or FSTVQ. The search process with the FSTVQ algorithm is identical to that of the LBG algorithm with the exception of using Eq. (4.36) and the proper η-schedule instead of hard clustering. The process starts with relatively small values of η that correspond to very fuzzy association. The level of randomness is reduced each time we increment η towards the optimal η^*. The issue of η scheduling is explained in the next section. It is worth mentioning that an update equation, similar to the one shown by Eq. (4.36), is derived in [97] following a DA approach for the special case of a labeled-state trellis coding system. In that brief paper, the authors use a forward-backward algorithm to solve for a set of probabilities similar to $\psi_n(l)$ in association with a dynamic programming algorithm to search for the minimum distortion path.

4.5 Performance Evaluation of FSTVQ

Several publications considered solving the problems associated with the LBG algorithm, which are mainly attributed to the descent to suboptimum local minima as well as the sensitivity to initialization, [15], [54], [109], [110], [111], [112] and [113]. Except [15] and [109], the developments in the former publications are based on different extensions to the original LBG algorithm. The appealing feature of the LBG algorithm is that it represents the practical implementation of the theoretical development of optimal quantization, which is referred to as Method I in [16]. In this section we shall apply the FSTVQ algorithm to solve the problems associated with the LBG algorithm in designing trellis waveform coding systems. The performance of the FSTVQ algorithm is tested through several examples as compared to the performances of both the STVQ and the VTVQ algorithms.

An important aspect that controls the performance of the FSTVQ algorithm is the choice of the relaxation range of η. The first thing to keep in mind is that it is desirable to give the FSTVQ algorithm the chance to iterate using the "optimal" η^*, which is experimentally chosen by optimizing the performance of the STVQ algorithm. Therefore, within the allowed number of iterations, we need to make sure that the η schedule converges at the end to η^*. Accordingly, depending on the total number of iterations and the speed of relaxation, we choose the initial η_{int}. By applying different forms of η-scheduling, experimental results showed a satisfactory performance using the following geometric schedule

$$\eta_n = \zeta \times \eta_{n-1}; \quad \eta_1 = \eta_{int}, n = 1, 2, \ldots, N_I \tag{4.37}$$

where ζ is a relaxation time-constant, and N_I is the total number of iterations performed by the search algorithm. The FSTVQ algorithm showed improved performance with slow η-scheduling at the early stages. This supported the choice of the schedule shown in Eq. (4.37), which provides slow updates for small η. An example of an η-schedule is presented in Fig. 4.3. Similar encounters of updating a Lagrange multiplier is necessary for DA applications, [57], [97], [56]. Nonetheless, these publications only comment on changing the corresponding Lagrange multiplier from 0 to ∞ without stating the procedure followed to perform this extreme transition. In particular, it is not clear how the authors in [97] increase the value of η to ∞ and still manage to use a forward-backward algorithm, which fails to function at extreme values of η, to compute a set of probability values.

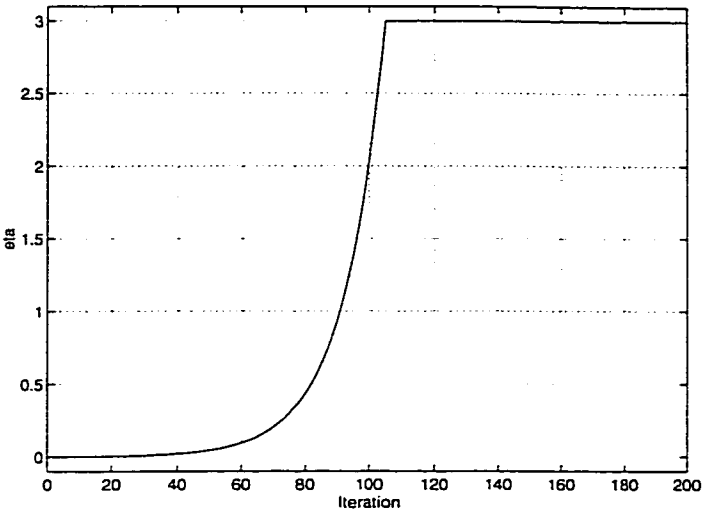

Figure 4.3: The η-schedule for the mixture Gaussian source: $\eta_{int} = 0.001$, $\zeta = 1.08$, $\eta^* = 3.0$.

4.5.1 Mixture Gaussian Source

The search process for optimal codebooks is substantially dependent on the complexity of the source of information. The word complexity is meant to denote the nonstationarity and the difficulty of the source statistical distribution model. In the framework of quantization, such sources are usually associated with complex distortion surfaces, which are hard to follow by conventional descent codebook search algorithms. A commonly used synthetic source, which models this situation is the mixture Gaussian source. The distribution of this source offers the attractive testing feature of a multi-local minima distortion surface [56], which can easily trap poorly-initialized locally-optimal search algorithms. The example involves encoding a mixture of eight randomly-displaced Gaussian distributions of equal variance. This source of information, which has a total variance of 31.401, is blocked into two-dimensional vectors for compression, as shown in Figs. 4.4, 4.5, and 4.6. As an example we use a trellis of $k = 1$, $\nu = 3$, thus $R = 0.5$ bps and the codebook size $= 8$, with FSTVQ, VTVQ and STVQ. The search processes over the trellis are initialized with the same codebook, which is randomly chosen from a 3000-sample training set. The different achieved minimum distortion points in the training phase are shown in Table 4.5. As expected STVQ and VTVQ, which operate with the LBG algorithm in the training phase, arrived at a similar minimum distortion configuration. The achieved minimum distortion point of the LBG-based algorithms is significantly (-0.69 dB) inferior to the solution provided by the FSTVQ algorithm.

Table 4.5: The rate 0.5 bps coding performance for the mixture Gaussian source. Results in the training phase represent the achieved minimum distortion points. The average performance of the different generated codebooks is categorized under testing.

	Training		Testing
	MSE	SNR (dB)	SNR $\pm$ CI (dB)
FSTVQ	9.758	5.076	4.873 ± 0.043
STVQ	11.426	4.390	4.290 ± 0.062
VTVQ	11.438	4.386	4.246 ± 0.064

It is clear that at a rate of 1 bit/vector, we have two optimal centers of mass that exhibit minimum moment of inertia. For this specific source of information the centroids are $c_1 = (-7.625, 3.875)$ and $c_2 = (2.35, -3.3)$. These centroids are indicated by an "o" in Figs. 4.4, 4.5 and 4.6. Accordingly, the optimal encoding codebook at this compression rate, is the one that equally clusters its components on the two optimal center of masses. In our case, the codebook size is 8, which means that 4 codevectors should be clustered at each center of mass. As shown in Fig. 4.4, the FSTVQ algorithm managed to escape local minima and located the optimal codewords at the "best" centers of mass at this particular compression rate. On the other hand, one of the codewords delivered by the STVQ and the VTVQ algorithms is trapped in a Gaussian cloud, as shown in Figs. 4.5 and 4.6. Furthermore, Figs. 4.5 and 4.6 show that the other "optimal" codewords are spread out at farther distances from the optimal centroids. The behavior of the FSTVQ algorithm in converging to a minimum distortion point is demonstrated in Fig. 4.7. The algorithm is applied for 200 iterations using the η-schedule shown in Fig. 4.3. Note that the convergence of the hard average distortion in Fig. 4.7, which is computed using Eqs. (4.31), (4.33) and (4.34), exhibits nondescending periods. This convergence behaviour is a feature of global optimization techniques (such as SA and DA), which allows the algorithm to search for other possibly lower energy configurations.

The optimal configurations provided by the different algorithms are tested using a 100 realizations of 1000 samples each. The average SNR with the corresponding 95% confidence intervals are shown under "Testing" in Table 4.5, which once again reflects the advantage of using the FSTVQ approach.

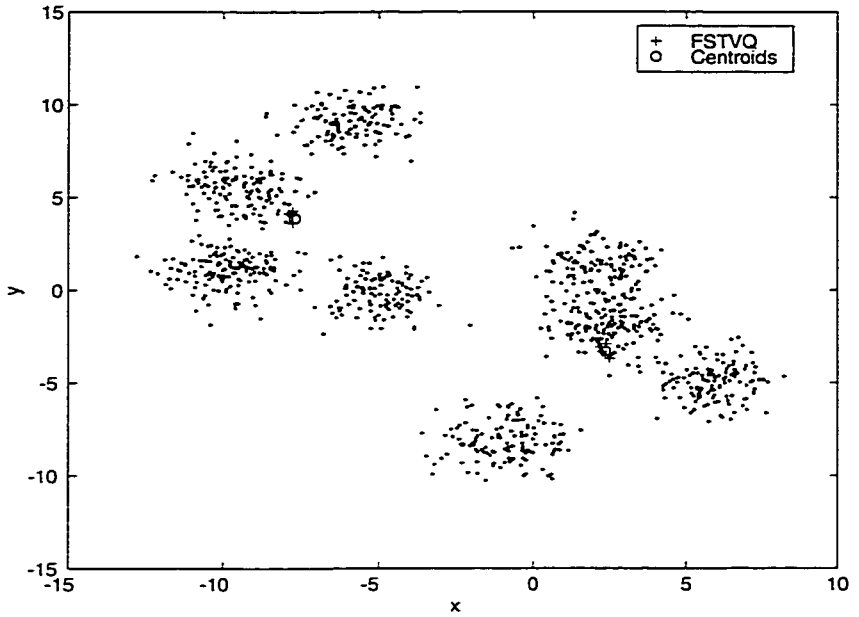

Figure 4.4: Two-dimensional mixture Gaussian source along with the optimal reproduction codewords using FSTVQ.

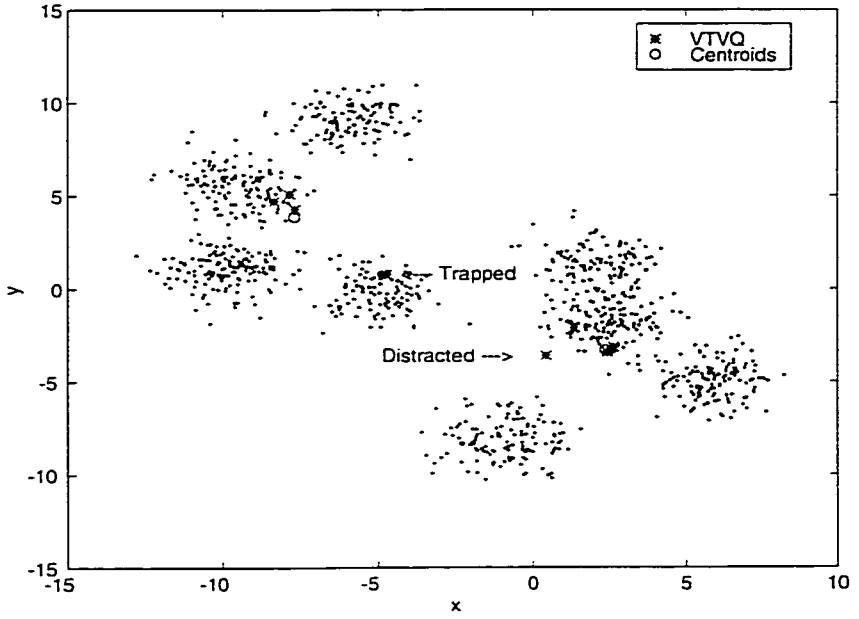

Figure 4.5: Two-dimensional mixture Gaussian source along with the optimal reproduction codewords using VTVQ.

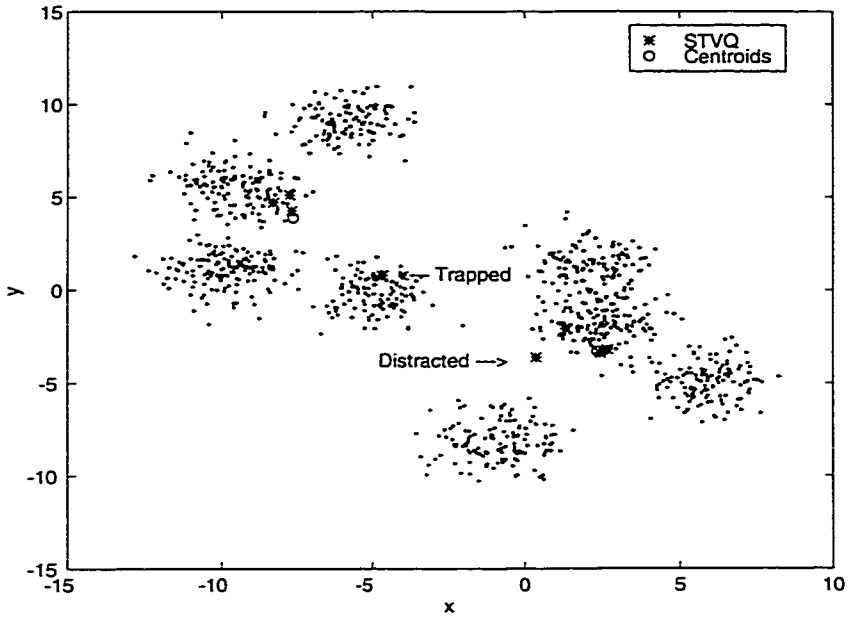

Figure 4.6: Two-dimensional mixture Gaussian source along with the optimal reproduction codewords using STVQ.

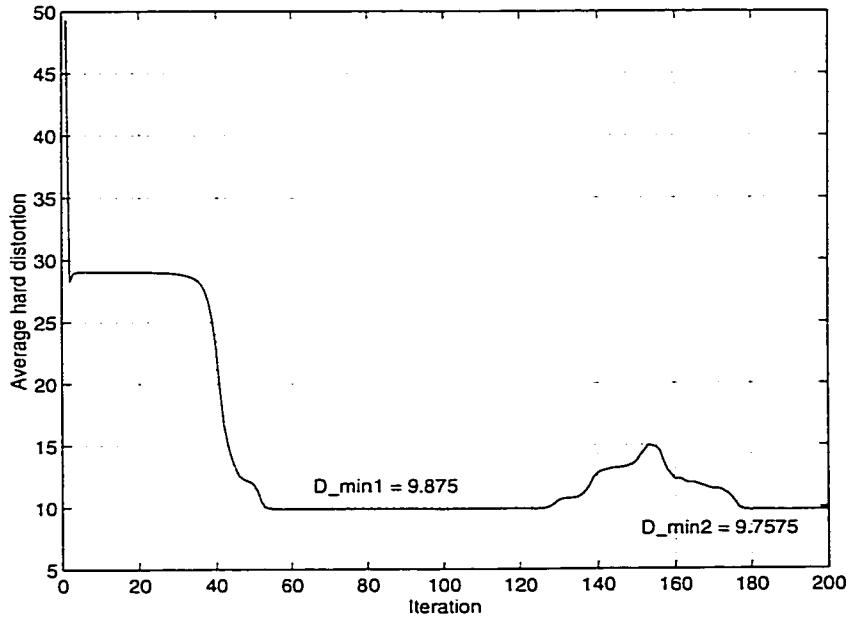

Figure 4.7: The convergence of the average hard distortion of reproduction using FSTVQ.

4.5.2 Second-Order Gauss-Markov Source

This section further shows the advantage of the FSTVQ algorithm in providing low distortion reproduction codebooks, irrespective of the initialization codebook or the training sequence length. Encoding is performed for a second-order Gauss-Markov source AR(2), which is defined by the following difference equation

$$y_n = 1.515y_{n-1} - 0.752y_{n-2} + z_n \tag{4.38}$$

where z_n represents a zero-mean Gaussian sequence. This source is often used to model sampled speech [14]. Appendix A provides a brief background about Gauss Markov sources as well as some rate-distortion results.

It is known that the distortion surface of Gauss-Markov sources does not have many local minima [57]. Therefore, it is very likely that a properly initialized LBG-algorithm might converge to a globally optimal solution using a long training sequence, which approximates the source distribution. Nonetheless, the statistical behavior of real signals is usually nonstationary and have an associated complex distortion surface. In these real applications, looking for a distribution-matched sequence and a proper initial codebook becomes rather challenging, which minimizes the odds of success of descent search algorithms. In order to simulate these real situations with the AR(2) source under consideration, we perform the search process using the different algorithms with relatively short training sequences and several initial codebooks.

On the other hand, given a particular compression rate, improved compression performance is expected with increased trellis complexity (constraint length). However, increasing the constraint length results in increasing the number of reproduction codewords. Searching for a larger set of optimal reproduction codewords is clearly more challenging. Not only the LBG algorithm becomes more sensitive to the initialization phase, but also it requires increased number of training samples to properly update the increased number of reproduction codewords. These limitations of the LBG algorithm are exposed in this case study by using several examples with different trellis structures. The different examples are conducted using two training sequences, one with 2000 samples and the other has only 1000 samples. The purpose of this approach is to test the sensitivity of the different algorithms to the training sequence lengths. Using each training set, the FSTVQ approach along with STVQ and VTVQ, which both use the LBG algorithm in the search process, are used with 15 initial codebooks that are chosen at random from the corresponding training sequence. Then, the lower energy configuration for each case is used to compute the average compression performance with the corresponding 95% confidence interval (CI).

The first example represents encoding the AR(2) source at a rate of 1 bps ($k = 1, L = 1$), for several trellis constraint lengths (ν). Figs. 4.8 and 4.9 show the SNR performance using the different initial codebooks with the 2000-sample training sequence. While Figs. 4.10 and 4.11 correspond to the results associated with the 1000-sample training sequence. Clearly these figures reflect the advantage of using FSTVQ in both providing significantly higher SNR levels and being almost independent of the initialization step. On the other hand, the LBG-based STVQ and VTVQ algorithms provided a similar performance, which was inferior to that of FSTVQ. Both STVQ and VTVQ failed to provide a consistent performance using the different initial codebooks. The performance fluctuation becomes more pronounced at higher constraint lengths. This is attributed to the fact that at higher ν the algorithms are still using the same small number of samples for training an increased number of reproduction codewords. Nonetheless, the two algorithms succeeded in matching the performance of the FSTVQ algorithm using some initial codebooks for the short constraint length of $\nu = 3$, as shown in Fig. 4.8(a). For this simple trellis structure (8 codewords), it happened that the initial codebooks are suitable for the LBG algorithm to converge to a better minimum distortion point.

Table 4.6 shows the average as well as the variance of the achieved minimum distortion points using the 15 initial codebooks. It is clear that for the different trellis constraint lengths, the FSTVQ approach provided higher average minimum distortion configurations. Furthermore, the LBG-based algorithms resulted in higher variance, which is due to the fluctuation in the achieved performance. This behaviour obviously reflects the dependency of the LBG algorithm on initial codebooks. On the other hand, reducing the length of the training sequence to 1000 samples forced the different algorithms to perform at higher average distortion points. This is clear in part (b) of Table 4.6 as well as Figs. 4.10 and 4.11. Note that reducing the training sequence length had more impact on the performance of the LBG algorithm with $\nu = 3$, compared to the other constraint lengths as shown in Table 4.6 parts (a) & (b). This is simply due to the fact that the performance of the LBG algorithm was already "poor" compared to that of FSTVQ using the training sequence length 2000. In other words, with 2000 training samples, the performance gap between the LBG-based algorithms and the FSTVQ algorithm is minimum at $\nu = 3$. This remark is depicted in Fig. 4.10(a) as compared to Fig. 4.8(a), where the LBG algorithm matches the FSTVQ performance for less number of times within the 15 trails.

The lowest energy configuration amongst the different initial codebooks is used to compute the average SNR performance with the 95% confidence interval. The results shown in Table 4.7 are computed using a 100 independent sequences of length 1000 samples of the AR(2) source. The superiority of the codebook generated by

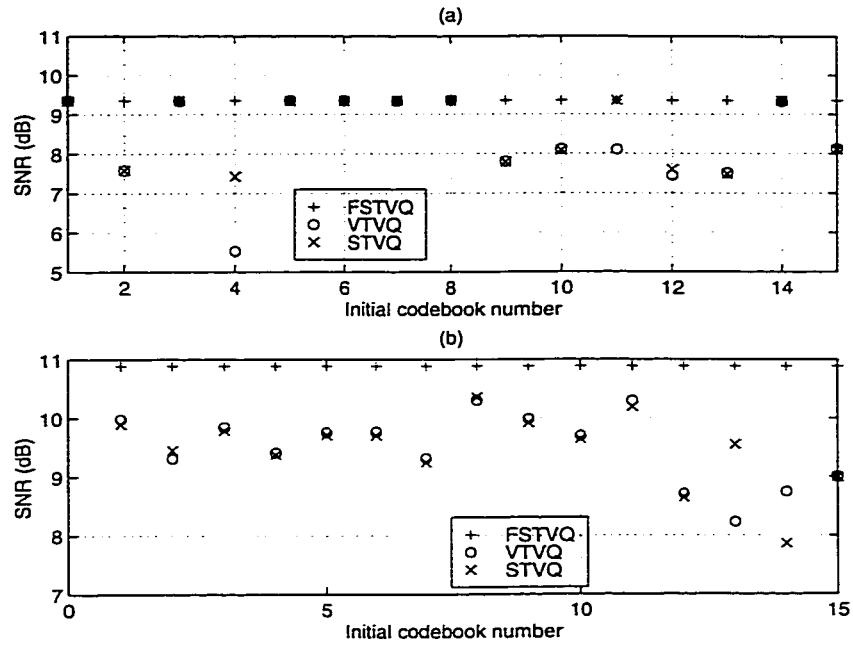

Figure 4.8: The SNR coding performance of the AR(2) source at a rate of 1 bps, $L = 1$, using 2000 training samples: (a) $\nu = 3$, (b) $\nu = 4$.

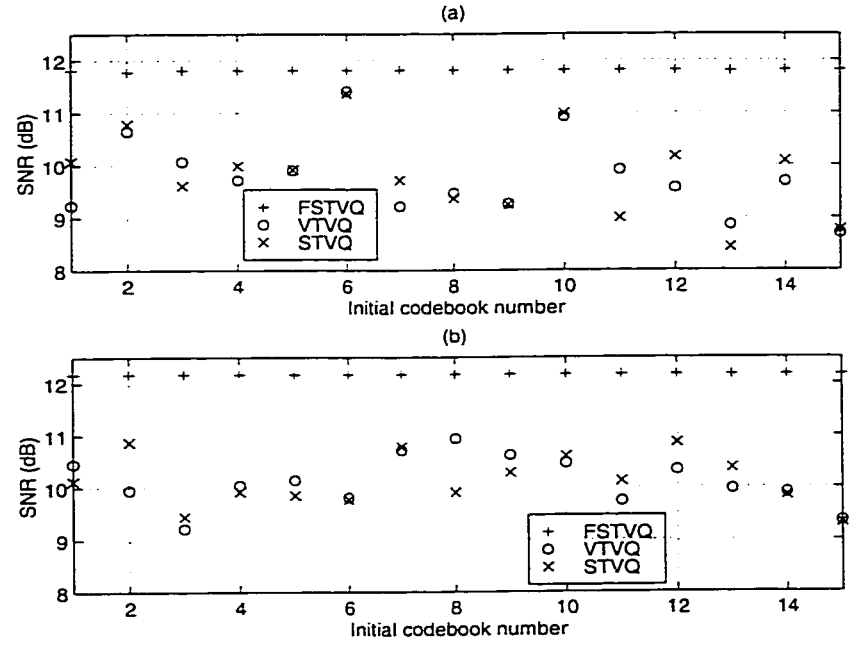

Figure 4.9: The SNR coding performance of the AR(2) source at a rate of 1 bps, $L = 1$, using 2000 training samples: (a) $\nu = 5$, (b) $\nu = 6$.

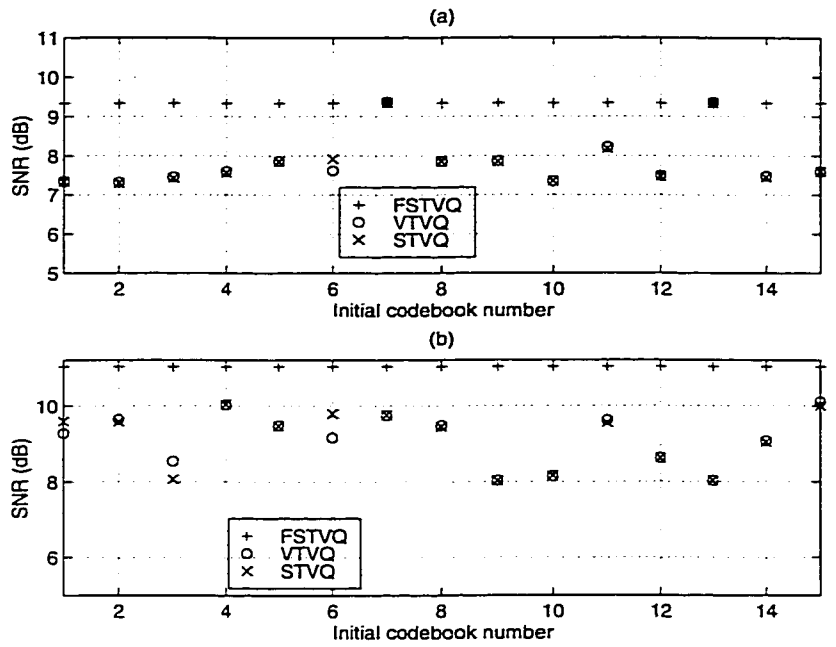

Figure 4.10: The SNR coding performance of the AR(2) source at a rate of 1 bps, $L = 1$, using 1000 training samples: (a) $\nu = 3$, (b) $\nu = 4$.

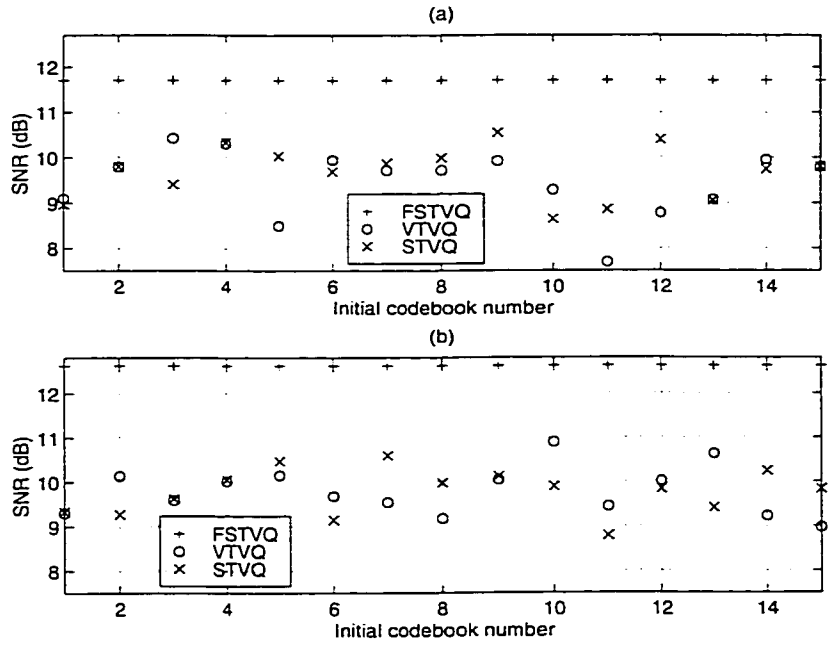

Figure 4.11: The SNR coding performance of the AR(2) source at a rate of 1 bps, $L = 1$, using 1000 training samples: (a) $\nu = 5$, (b) $\nu = 6$.

Table 4.6: The mean (μ) and the variance (σ^2) of the minimum distortion points (as SNR in dB) generated using the different algorithms with the AR(2) source. The compression rate is 1 bps, $k = 1$ and $L = 1$:(a) using 2000 traning vectors, (b) using 1000 training vectors, (c) using 4000 training vectors.

		STVQ		VTVQ		FSTVQ	
	ν	μ	σ^2	μ	σ^2	μ	σ^2
(a)	3	8.592	0.736	8.377	1.246	9.355	0.000
	4	9.491	0.392	9.492	0.370	10.888	0.000
	5	9.836	0.662	9.775	0.563	11.810	0.000
	6	10.139	0.236	10.111	0.234	12.171	0.000
(b)	3	7.857	0.430	7.851	0.433	9.338	0.000
	4	9.145	0.569	9.135	0.491	11.028	0.000
	5	9.677	0.336	9.467	0.539	11.707	0.000
	6	9.775	0.258	9.787	0.299	12.917	0.000
(c)	6	10.563	0.392	10.723	0.467	12.243	0.000

FSTVQ is depicted in Table 4.7, where at $\nu = 6$ the advantage is approximately 1.8 dB. Furthermore, in Table 4.7 we demonstrate the effect of reducing the number of training samples on the LBG algorithm, especially at higher values of ν. Clearly by choosing the minimum distortion configuration of the LBG algorithm we present the best achievable performance of this algorithm using 15 different initial codebooks. Nonetheless, the average performance of the LBG algorithm would have been severely deteriorated, had we chosen other higher distortion configurations. This is true, since the highest distortion configuration of the FSTVQ algorithm in the different examples are significantly lower than those evaluated by using the LBG algorithm. It is worth mentioning at this point that the distortion rate function of this source at 1 bps is $D(1) = 14.96$ dB [14].

Note that the performance of the TVQ process should improve as we increase the constraint length of the compression process. Nonetheless, due to the increased number of codewords and the relatively small number of training vectors (2000 or 1000), the generated codebooks using the LBG algorithm at $\nu = 6$ failed to outperform the optimal codebooks associated with $\nu = 5$, as shown in Table 4.7. In order to show that this is strictly related to the length of the training sequence, we perform the search for the optimal codebooks at $\nu = 6$ using 4000 training vectors. The results are shown in part (c) of Tables 4.6 and 4.7. These results clearly show the achieved improved performance compared to the results that correspond to 1000 and 2000 training vectors. Once again, FSTVQ outperformed the LBG-based algorithms at this relatively higher training sequence length.

Table 4.7: The rate 1 bps average coding performance for the AR(2) source; $k = 1$, $L = 1$. The values are listed as SNR $\pm$ CI in (dB): (a) using 2000 training vectors, (b) using 1000 training vectors, (c) using 4000 training vectors.

	ν	STVQ	VTVQ	FSTVQ
(a)	3	9.118 ± 0.113	9.111 ± 0.109	9.117 ± 0.110
	4	10.159 ± 0.090	10.124 ± 0.067	10.783 ± 0.065
	5	11.178 ± 0.065	11.167 ± 0.065	11.729 ± 0.072
	6	10.135 ± 0.194	10.305 ± 0.072	12.1126 ± 0.064
(b)	3	9.105 ± 0.172	9.106 ± 0.171	9.102 ± 0.169
	4	10.102 ± 0.093	10.104 ± 0.094	10.656 ± 0.133
	5	10.209 ± 0.096	9.984 ± 0.112	11.706 ± 0.070
	6	9.688 ± 0.112	9.596 ± 0.128	11.905 ± 0.052
(c)	6	11.401 ± 0.095	11.295 ± 0.087	12.221 ± 0.070

Although Figs. 4.8, 4.9, 4.10 and 4.11 show some discrepancy between the performances of STVQ and VTVQ with some initial codebooks, these results do not reflect the general behavior of the two algorithms, since we are using a relatively short training sequence. The training sequence in our case is not capable of representing the source distribution, which means that these results do not reflect the average performance of the two algorithms presented in section 4.3.

To further show the behavior of the FSTVQ algorithm with other trellis structures, we perform encoding at a rate of 2 bps. At this rate, the source has a theoretical distortion value $D(2) = 21.64$ dB. The trellis in this case is constructed with $k = 2$ and $L = 1$. The performance results using 15 different initial codebooks with the two training sequences are shown in Figs. 4.12 and 4.13, for $\nu = 2, 3$. The same observations are demonstrated in these figures; the FSTVQ algorithm managed to arrive at lower energy configurations using a short training sequence without being sensitive to initial codebooks. Furthermore, due to the increased complexity of this encoding trellis as compared to the first example ($k = 1$), the LBG algorithm failed to match the performance of the FSTVQ algorithm using any of the initial codebooks. Table 4.8 represents the average results associated with Figs. 4.12 and 4.13. Note that due to the increased complexity of the trellis, and unlike the results of Table 4.6, some of the σ^2 under FSTVQ have non-zero values. Nonetheless, these values are significantly smaller that the values associated with the performance of the LBG algorithm. To demonstrate the average performance of the best codebooks generated by the different algorithms, we perform encoding using 100 different sequences of 1000 sample each. The average SNR performance with the 95% CI is shown in Table 4.9. The advantage of using the codebook generated by the FSTVQ approach reaches 3.08 dB for the rate 2 bps with $\nu = 3$ using 1000 training samples.

In particular, this example shows the indifference of the FSTVQ algorithm to using only 1000 training samples as compared to the LBG-based algorithms.

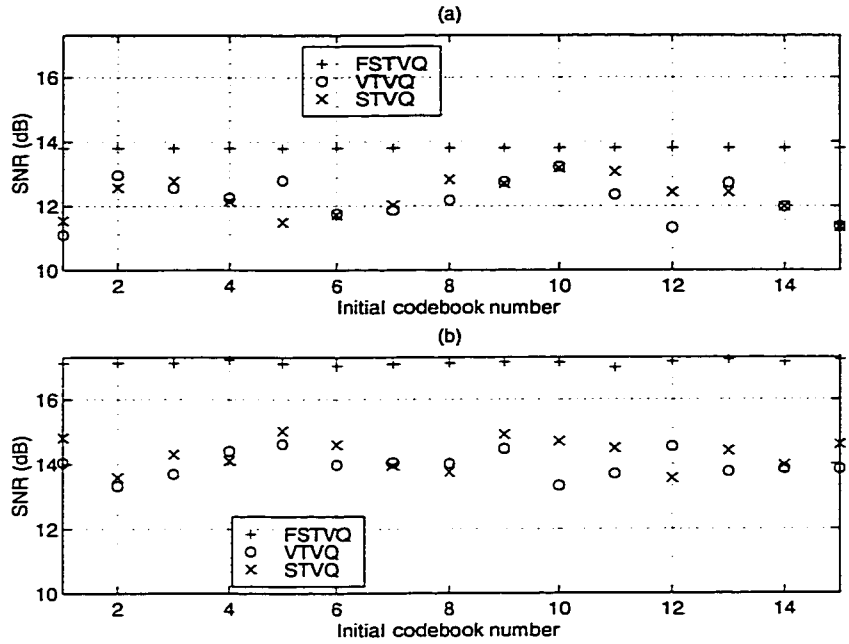

Figure 4.12: The SNR coding performance of the AR(2) source at a rate of 2 bps, $L = 1$, using 2000 training samples: (a) $\nu = 2$, (b) $\nu = 3$.

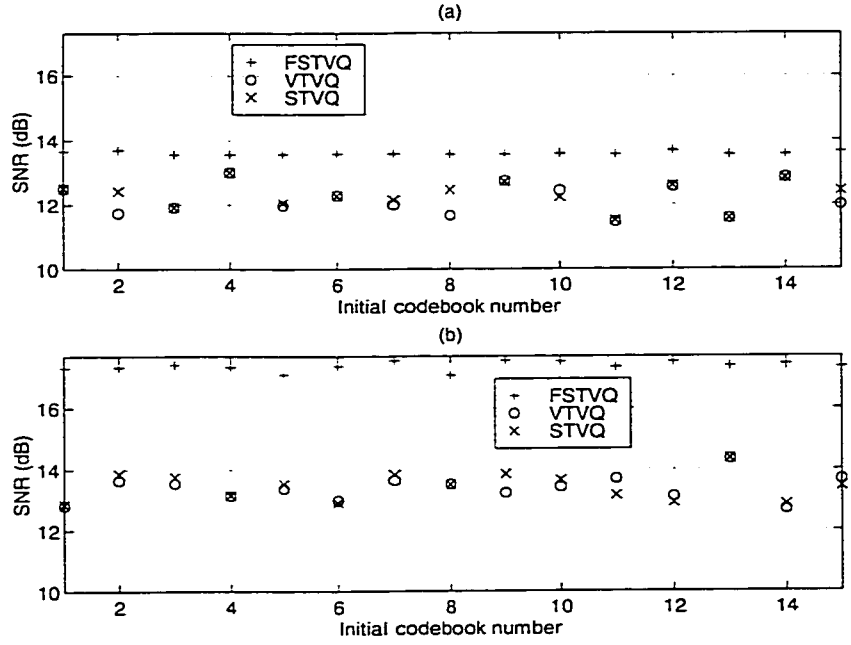

Figure 4.13: The SNR coding performance of the AR(2) source at a rate of 2 bps, $L = 1$, using 1000 training samples: (a) $\nu = 2$, (b) $\nu = 3$.

Table 4.8: The mean (μ) and the variance (σ^2) of the minimum distortion points (as SNR in dB) generated using the different algorithms with the AR(2) source. The compression rate is 2 bps, $k = 2$ and $L = 1$: (a) using 2000 training vectors, (b) using 1000 training vectors.

		STVQ		VTVQ		FSTVQ	
	ν	μ	σ^2	μ	σ^2	μ	σ^2
(a)	2	12.283	0.342	12.214	0.410	13.800	0.000
	3	14.322	0.227	13.982	0.161	17.130	0.006
(b)	2	12.310	0.179	12.177	0.232	13.589	0.003
	3	13.440	0.205	13.388	0.171	17.377	0.021

Table 4.9: The rate 2 bps average coding performance for the AR(2) source; $k = 2$, $L = 1$.The values are listed as SNR $\pm$ CI in (dB): (a) using 2000 training samples, (b) using 1000 training samples.

	ν	STVQ	VTVQ	FSTVQ
(a)	2	13.150 ± 0.050	13.135 ± 0.068	13.572 ± 0.099
	3	14.263 ± 0.099	14.026 ± 0.061	16.566 ± 0.055
(b)	2	12.604 ± 0.086	12.639 ± 0.078	13.266 ± 0.082
	3	13.302 ± 0.053	13.257 ± 0.053	16.380 ± 0.036

4.5.3 Image Compression

The objective of this section is to show that the FSTVQ approach is also beneficial in the area of image compression. Furthermore, this section serves as a good example for searching for optimal codebooks over complex distortion surfaces, which are known to be associated with vector-based image compression. We shall consider several cases with different trellis structures and compression rates. The examples involve using two different training sets one with 2 images and the other one with 5 training images as well as a test image that is not used in either of the training sets [114], [115], [116] and [117]. Clearly increasing the number of training images reflects into improving the performance of the optimal codebook when applied to encoding the test image. The training sets consists of 256×256 gray-scale images represented by 8-bit pixels. The test image, on the other hand, is the 512×512 gray-scale Lenna image. The search for the optimal codebook is independently repeated for 15 times, each time with a different initial codebook chosen uniformly at random from [0 255].

The first example considers using a trellis with $k = 1$ and several constraint lengths ν. Compression is performed using 4×4 image blocks producing a vector dimension $L = 16$. Accordingly, the compression rate $R = k/L = 0.0625$ bit per pixel (bpp). The performance measure is the peak signal to noise ratio (PSNR) defined in appendix C. The results shown in Figs. 4.14 and 4.15 represent the PSNR performance of the FSTVQ algorithm compared to the LBG-based STVQ and VTVQ search algorithms, using 2 training images at a rate of 0.0625 bpp. It is clear that FSTVQ provides lower energy configurations while being significantly less sensitive to the initialization process.

Note that unlike the case of the AR(2) source, the dependency of the LBG algorithm in this particular case on initial conditions decreased with increased values of ν. This is clear from Figs. 4.14 and 4.15 and Table 4.10 (a), where we show the average as well as the variance of the achieved PSNR values using the 15 different initial codebooks. It is clear that the fluctuation in performance is reduced as ν increases, this observation is more obvious by comparing the computed values of σ^2 in Table 4.10(a). This behaviour is simply attributed to the fact that higher values of ν corresponds to larger values of codebook size. At the same time the initial value for a pixel in a codeword is to be chosen uniformly at random from [0 255]. Accordingly, and due to this limited number of initial choices, the difference between the geometrical map of the different initial codebooks naturally becomes less significant as the codebook size increases. Nevertheless, the performance of the FSTVQ approach with images is similar to the AR(2) source; i.e., the dependency on initial codebooks increased with increased values of ν. This is simply due to the nature of FSTVQ, which is expected to be completely independent of initialization.

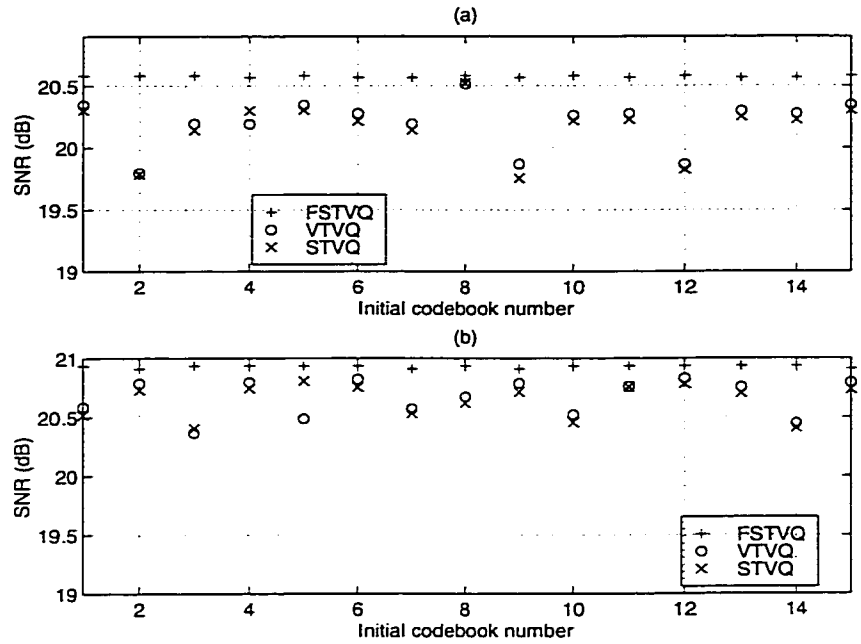

Figure 4.14: The rate 0.0625 bpp PSNR coding performance in (dB) using 2 training images: (a) $\nu = 3$, (b) $\nu = 4$.

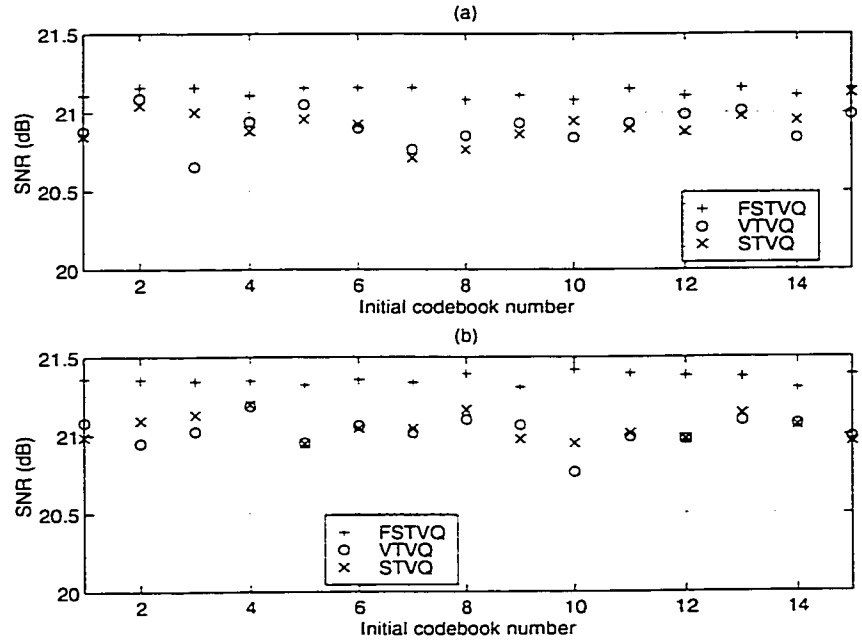

Figure 4.15: The rate 0.0625 bpp PSNR coding performance in (dB) using 2 training images: (a) $\nu = 5$, (b) $\nu = 6$.

Table 4.10: The mean (μ) and the variance (σ^2) of the minimum distortion points (as PSNR in dB) generated using the different algorithms with the training images. The compression rate is 0.0625 bps, $k = 1$ and $L = 16$: (a) using 2 training images, (b) using 5 training images.

		STVQ		VTVQ		FSTVQ	
	ν	μ	σ^2	μ	σ^2	μ	σ^2
(a)	3	20.168	0.047	20.202	0.041	20.575	0.000
	4	20.646	0.020	20.667	0.024	20.929	0.000
	5	20.920	0.011	20.912	0.013	21.131	0.001
	6	21.045	0.007	21.021	0.009	21.359	0.001
(b)	3	19.755	0.079	19.766	0.080	20.317	0.000
	4	20.361	0.049	20.352	0.079	20.733	0.000
	5	20.759	0.008	20.742	0.042	20.959	0.001
	6	20.996	0.009	20.983	0.008	21.171	0.001

Table 4.11: The rate 0.0625 bpp PSNR coding performance in dB for the Lenna image, $k = 1$ and $L = 16$: (a) using 2 training images, (b) using 5 training images.

	ν	STVQ	VTVQ	FSTVQ
(a)	3	20.992	21.002	21.070
	4	21.230	21.296	21.739
	5	21.680	21.677	22.011
	6	21.909	21.844	22.160
(b)	3	21.377	21.344	21.454
	4	22.167	22.197	22.630
	5	22.510	22.370	22.867
	6	22.787	22.820	23.166

However, more complex trellis structures tend to slightly degrade the performance of the FSTVQ algorithm due to the associated increased complexity of the distortion surface. The lowest distortion configuration for each value of ν is used to encode the Lenna image. The PSNR results are shown in Table 4.11(a), where we demonstrate the advantage of using FSTVQ for the different constraint lengths.

In the second part of the first case, we are still considering the compression at a rate $R = 0.0625$ bpp ($k = 1, L = 16$), however, using 5 training images. The highest achieved PSNR values in the training phase are plotted in Figs. 4.16 and 4.17. As shown in Fig. 4.16(a) , using the 8th initial codebook the STVQ and the VTVQ algorithms succeeded in matching the performance of the FSTVQ algorithm, which provided a consistent performance using the different initial codebooks. Clearly, the fact that we are using a relatively long training sequence made it easier for the LBG-based algorithms to converge to a lower energy configuration, which was simply coincidental. On the other hand, the FSTVQ algorithm proved to provide relatively better PSNR results with more complex trellis structures, as shown in Fig. 4.17. The dependency (fluctuation) of the performance of the LBG on initial conditions is clearly demonstrated in Table 4.10(b). Similar to the observations made earlier, it is clear in this case too that the fluctuation measured by the values of σ^2 gets smaller with increased values of ν.

Table 4.11(b) shows the results of encoding the Lenna image using the lower distortion codebook for the different algorithms and constraint lengths. Comparing the performance of the different algorithms in both the training phase and the testing phase, we notice a clear advantage of the codebook generated using FSTVQ, especially at $\nu = 4$ in Table 4.11(b). Although the different algorithms arrived at a comparable low distortion point using initial codebook number 12, as shown in Fig. 4.16(b), the corresponding optimal codebooks should not necessarily be expected to have similar components. In this particular example, the FSTVQ lowest distortion point is achieved using initial codebooks $(1, 4, 5, 6, 8, 10, 11)$. Accordingly, when encoding an external test image, these different initial codebooks are believed to provide different performance.

The other case study considers encoding at a rate of 0.25 bpp. For this system we use a trellis of $k = 2$ and constraint lengths $\nu = 2, 3$, as well as 4×2 image blocks, thus a vector dimension $L = 8$. To show the superiority of FSTVQ with short training sequences, we only consider the 2-image training set. Fig. 4.18 shows the PSNR performance results using 15 uniformly distributed random initial codebooks. Note that at this relatively more complex trellis, the FSTVQ approach showed some dependency on the initialization step, as shown in Fig. 4.18 and

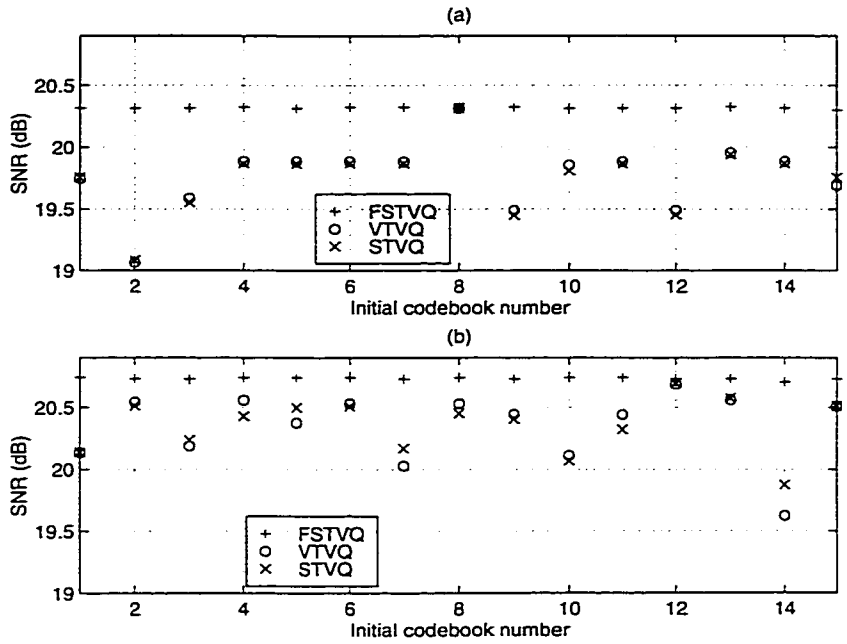

Figure 4.16: The rate 0.0625 bpp PSNR coding performance in (dB) using 5 training images: (a) $\nu = 3$, (b) $\nu = 4$.

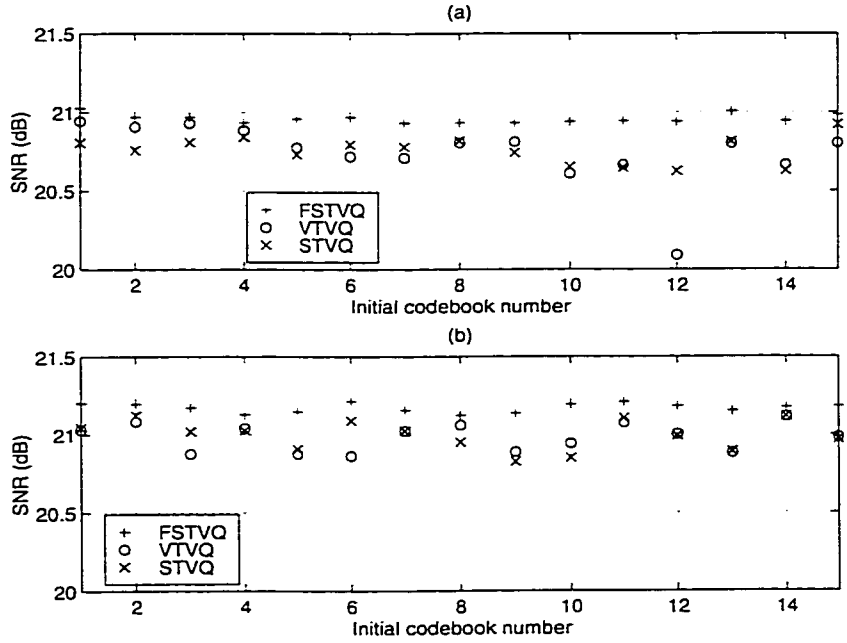

Figure 4.17: The rate 0.0625 bpp PSNR coding performance in (dB) using 5 training images: (a) $\nu = 5$, (b) $\nu = 6$.

Table 4.12: The mean (μ) and the variance (σ^2) of the minimum distortion points (as PSNR in dB) generated using the different algorithms with 2 training images. The compression rate is 0.25 bps, $k = 2$ and $L = 8$.

	STVQ		VTVQ		FSTVQ	
ν	μ	σ^2	μ	σ^2	μ	σ^2
2	22.595	0.013	22.565	0.020	22.772	0.002
3	23.175	0.009	23.157	0.008	23.377	0.001

Table 4.12. However, the FSTVQ performance is still better than that of the LBG-based algorithms. The improved performance is further supported in in Table 4.13, which represent the PSNR performance of encoding the Lenna image using the best generated codebooks in the training phase.

To show the impact of the PSNR improvement on the image subjective quality, Figs. 4.20 to 4.25 show the reproduced images associated with the results of Table 4.13. In general, MSE-based VQ of images suffer from what is known as the staircase effect in image areas that include edges [114]. Several studies have successfully treated this VQ compression deficiency, we mention in this context subband VQ of images [115], addressed-VQ [117] and classified VQ [114, 116]. The MSE compression process for VQ and TVQ is similar in the sense that in both cases the reproduction codeword at any intermediate step in the process is the one with the minimum Euclidean distance. Therefore, it is expected to face the same staircase effect problem with TVQ. Nonetheless, in this case study we are interested in showing the relative improvement resulting from applying the different codebook search algorithms with image sources. Therefore, we will not consider applying staircase-proof techniques, which have been thoroughly investigated in literature.

The original Lenna image is shown in Fig. 4.19. The images in Figs. 4.20, 4.21 and 4.22 represent the results with a trellis constraint length $\nu = 2$, where it is clear that the PSNR improvement is reflected into enhancing the quality of the corresponding reproduced image. In particular, higher PSNR values resulted into improving the quality of high frequency values like the feather area. The same conclusions apply for Figs. 4.23, 4.24 and 4.25, which are associated with $\nu = 3$.

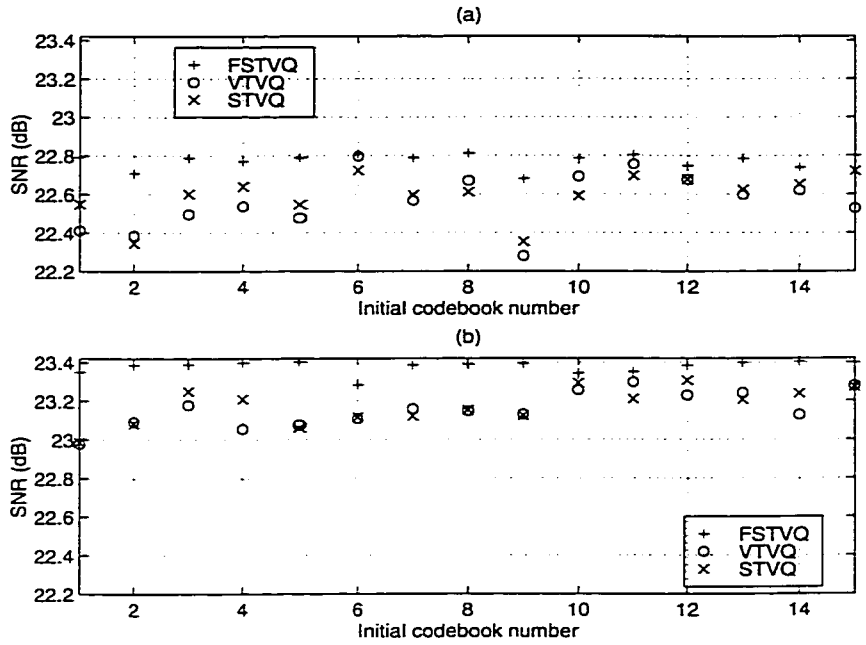

Figure 4.18: The rate 0.25 bpp PSNR coding performance in (dB) using 2 training images: (a) $\nu = 2$, (b) $\nu = 3$.

Table 4.13: The rate 0.25 bpp PSNR coding performance in dB for the Lenna image using 2 training images.

ν	STVQ	VTVQ	FSTVQ
2	23.901	24.047	24.614
3	24.738	24.693	25.065

Figure 4.19: The original Lenna image.

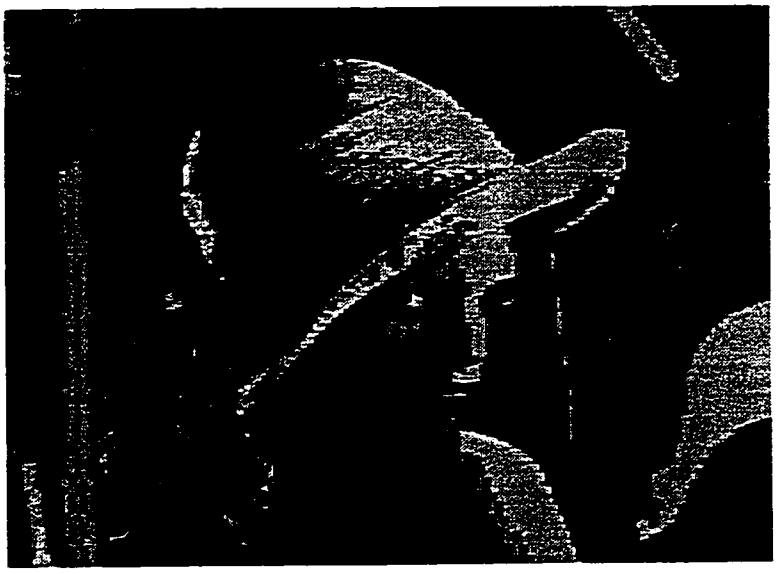

Figure 4.20: The rate 0.25 bpp reconstructed image with FSTVQ, $\nu = 2$ and PSNR = 24.614.

Figure 4.21: The rate 0.25 bpp reconstructed image with STVQ, $\nu = 2$ and PSNR = 23.901.

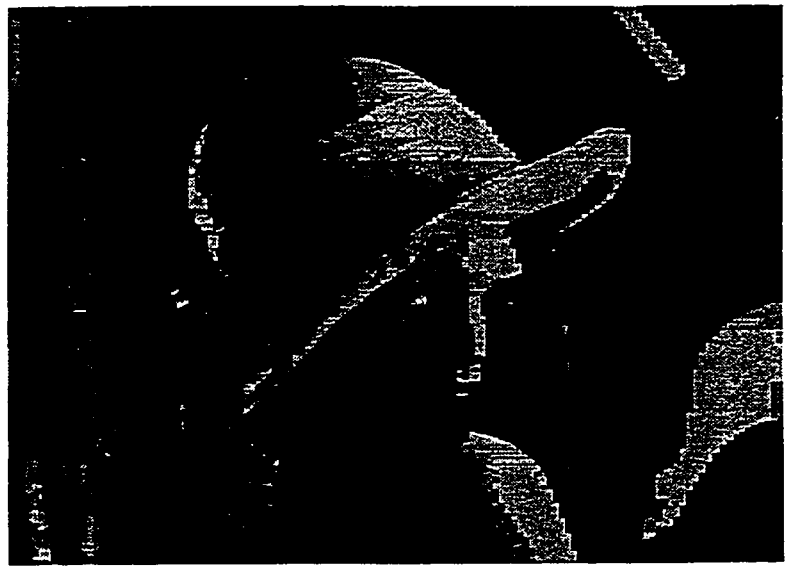

Figure 4.22: The rate 0.25 bpp reconstructed image with VTVQ, $\nu = 2$ and PSNR = 24.047.

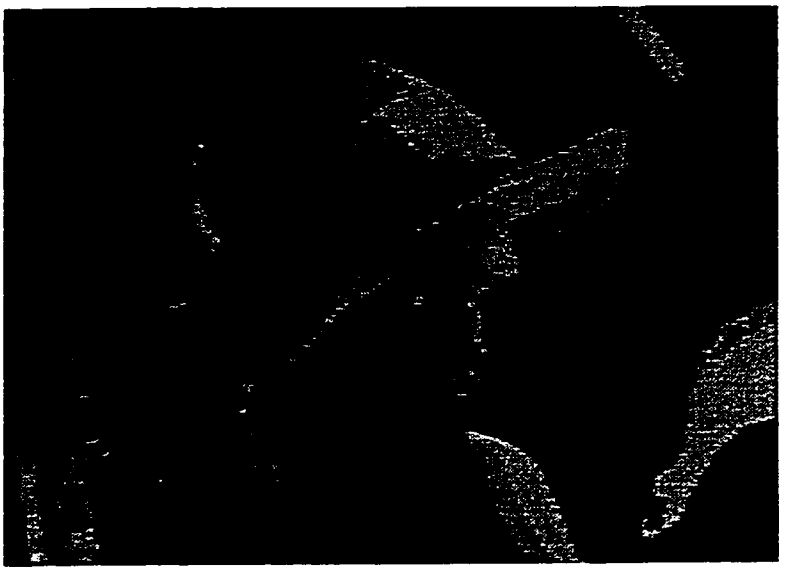

Figure 4.23: The rate 0.25 bpp reconstructed image with FSTVQ, $\nu = 3$ and PSNR = 25.065.

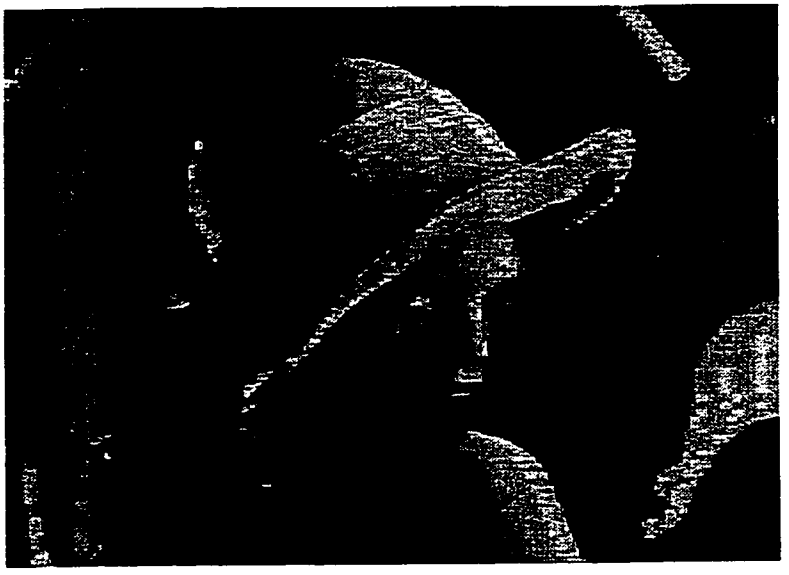

Figure 4.24: The rate 0.25 bpp reconstructed image with STVQ, $\nu = 3$ and PSNR = 24.738.

4.6 Summary

This chapter introduced a novel approach, which used the source-channel coding analogy to implement the symbol-MAP algorithm for data compression. The new soft compression algorithm (STVQ) delivers a sequence of minimum distortion symbols, which is commensurate to the minimum distortion sequence generated by the Viterbi algorithm. Both the STVQ algorithm and the Viterbi algorithm performed within the same rate-distortion proximity with memoryless Gaussian and Gauss Markov sources. Even though the STVQ algorithm is more complex than the Viterbi algorithm, through its distortion-related soft information, the new algorithm offers new degrees of freedom for the design of reliable compression systems. We showed that the complexity of this forward-backward algorithm can be cut down to roughly 2-4 times the complexity of the Viterbi algorithm.

In the second part of this chapter we considered extending the STVQ algorithm by using the distortion-related reliability information to provide a fuzzy soft compression approach (FSTVQ). In the process of deriving the FSTVQ algorithm, we showed th close relationship between the functionality of Blahut algorithm and the STVQ algorithm. The two algorithms use similar association probabilities and lagrange multipliers to approach the sought for minimum distortion-rate point. This interpretation led to proposing the minimization of a soft distortion measure, which culminated into deriving the FSTVQ codewords update rule to replace the LBG-based STVQ codebook design technique. The new approach alleviated the problems associated with the LBG search algorithm as applied to TVQ. The FSTVQ search process provided lower distortion optimal codebooks that are significantly independent of the initialization process using short training sequences. Due to the nonstationarity of real data sources, it is not always preferable or even possible to have a suitable long training sequence. FSTVQ showed significant improvement over the LBG algorithm using short training sequences, which makes the new approach an attractive choice for real applications. The performances of both the FSTVQ-algorithm and the LBG-based algorithms were compared using a mixture Gaussian source, an AR(2) Gauss-Markov source as well as still images.

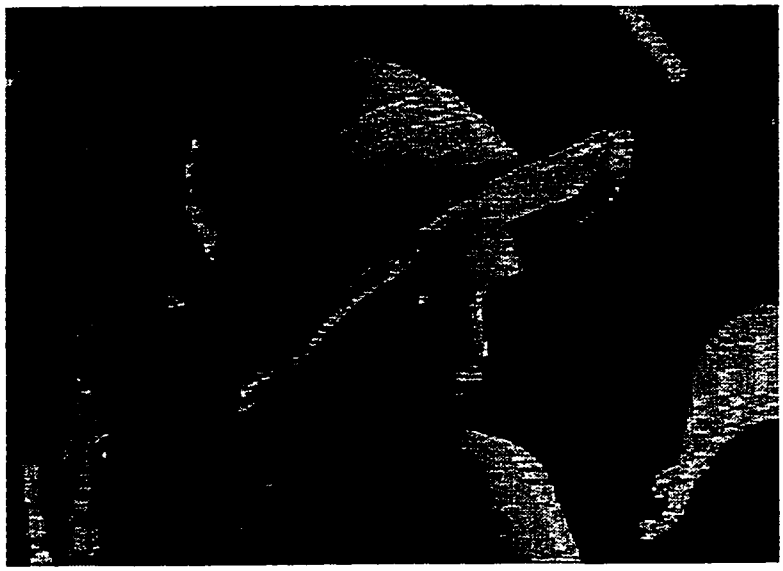

Figure 4.25: The rate 0.25 bpp reconstructed image with VTVQ, $\nu = 3$ and PSNR = 24.693.

Chapter 5

Channel-Optimized Soft Trellis Waveform Coding

In [46], Shannon shows that as long as the source entropy is less than the channel capacity it is feasible to achieve an arbitrarily low error probability using separable source and channel coding systems. Although most of existing communication systems apply the separable Shannon coding theory, tandem source and channel coding systems only provide suboptimal performance. The suboptimality is mainly attributed to the fact that the independent design of either system is based on the ideal performance of the other one, which is practically rather elusive. In order to explain this fact, consider the design of a channel coding system that is based on ideal source coding. First, ideal source compression systems are characterized by delivering memoryless and uniformly-distributed sequences of finite-alphabet symbols. In other words, they produce uncorrelated sequences of symbols that exhibit maximal amount of information over the induced alphabet set; i.e., redundancy-free sequences. However, practically any kind of nonstationarity in the source statistics or suboptimality in the compression approach always reflects into residual redundancy at the source encoder output. Therefore, a channel coding system that is designed without considering the residual redundancy at its input is expected to fail to provide the anticipated level of error protection in practical situations.

On the other hand, the design of channel-independent source compression systems assume error-free transmission. The implication of this design approach represents

the loss of some of the fidelity of reproduction over real noisy channels. The loss can be significantly reduced by introducing an error correcting code, which practically reduces the error rate over the channel to zero. However, the channel coding system imposes extra complexity, delay and bandwidth requirements on the communications system. In order to solve this issue of suboptimality several publications considered a joint source and channel coding approach. Generally speaking, joint source and channel coding systems function to properly exploit the trade-off between fidelity of reproduction and error protection over channels within fixed transmission bandwidths.

Several studies considered the effect of channel noise and the design of channel optimized scalar quantizers [118, 119, 23]. Furthermore, the significance of vector quantization systems in different speech and image compression applications motivated research efforts into developing channel optimized vector quantizers (COVQ) [59, 120, 121, 24], [25, 122], [21, 22]. Applying COVQ proved to provide lower distortions of reproduction over the noisy channel by constraining the codebook design and the encoding process by the distribution of the channel noise. Encoding is performed by transmitting the codeword index that produces the lowest average distortion over the channel distribution, rather than transmitting the index of the codeword closest to the source output vector. On the other hand, the codebook search process is performed through minimizing an average distortion measure, with averaging performed over both the source and the channel statistics.

Channel-optimized trellis vector quantization (CO-TVQ) is a natural extension to the basic principles of COVQ. The theory and the application of CO-TVQ systems is presented in [21] and [22]. The CO-TVQ system design approach presented in [22] is based on employing the LBG algorithm along with an initial codebook, a "long" training sequence and a search algorithm. However, for a practical range of channel bit error rates, the LBG algorithm suffers from the same limitations of the noiseless channel case of simply being a local descent algorithm. Motivated by the performance of the FSTVQ approach presented in the last chapter, we provide in this chapter the necessary development for a channel-optimized FSTVQ codebook search algorithm (CO-FSTVQ). Experimental results show the superiority of the new CO-FSTVQ approach in providing robust channel-optimized trellis codebooks, which are capable of providing competing distortion performance under channel-matched and mismatched conditions. Simulations are conducted using first and second order Gauss-Markov sources using several trellis structures, different initial codebooks as well as short training sequences.

5.1 Background and Problem Statement

Due to the close relationship between regular VQ and TVQ, it is always more insightful to look into a TVQ problem from a VQ perspective. This section provides the notation, the problem statement as well as the necessary background for COVQ and CO-TVQ as an introduction for the generalization of FSTVQ to the counterpart channel-optimized system, which is presented in the next section. The source considered in this chapter is a discrete-time continuous-amplitude source $\{y_n\}_{n=0,1,2,\ldots}$, which is blocked into L-dimensional vectors as

$$Y_n = [y_{nL}, y_{nL+1}, \cdots, y_{nL+L-1}]$$

Furthermore, the quantizers use a codebook denoted by $\mathcal{C} = \{C_0, C_1, \ldots, C_{K-1}\}$, with K representing the codebook cardinality.

5.1.1 Channel Optimized Vector Quantization

The pioneer work of Farvardin in [24] set the basics for COVQ systems. In order to introduce COVQ, we start by defining an encoding probability-like variable similar to the one defined in Eq. (4.32) as

$$\pi_n(i) = \begin{cases} 1, & Y_n \in S_i \\ 0, & \text{otherwise} \end{cases} \tag{5.1}$$

where S_i represents the ith encoding set. Note that the definition in Eq. (5.1) represents a hard association rule. Then, similar to the conventional VQ design approach, the design of COVQ is based on minimizing an average distortion of reproduction, however, averaging is over both the source and the channel distributions [24, 25]

$$D = \frac{1}{N_T} \sum_{n} \sum_{i=0}^{K-1} \sum_{j=0}^{K-1} \pi_n(i) P_{j|i} d(Y_n, C_j) \tag{5.2}$$

where $d(Y_n, C_j)$ represents the instantaneous distortion between the vector Y_n and the codeword C_j. Furthermore, $P_{j|i}$ is the probability that the symbol j is received given that i was sent over the channel, and N_T is the length of the training sequence. For the special case of a squared error distortion, Eq. (5.2) reduces to the sum of the distortion introduced by VQ over the noiseless channel (D_S) and the contribution of the noisy channel to the overall distortion (D_C) [24]. For this particular case, it is possible to individually minimize the two distortion components. Clearly, minimizing the D_S represents a regular VQ design problem, which we refer to in this chapter as source-matched quantization. Minimizing the distortion associated

with the channel noise D_C is solved by following an index assignment (IA) technique [59, 120, 121, 24]. Nonetheless, as demonstrated in [97], solving for a COVQ using an IA approach is locally suboptimal, since it is substantially dependent on the noiseless VQ design.

The other approach that guarantees locally optimal COVQ structures involves minimizing the average distortion given by Eq. (5.2). The design procedure is similar to that of the noiseless VQ case. In this case the Nearest Neighbor Condition (NNC), which represents the encoding rule, is given by [25, 24, 122]

$$S_i = \left\{ Y_n : \sum_{j=0}^{K-1} P_{j|i}\, d(Y_n, C_j) < \sum_{j=0}^{K-1} P_{j|l}\, d(Y_n, C_j) \right\} \quad l, i = 0, 1, ..., K-1 \qquad (5.3)$$

Note that the encoding rule takes care of the fact that the COVQ output is passing through the noisy channel by delivering the index, which is most likely to produce minimum distortion given the distribution of the channel noise. The second condition of optimality resembles the solution to the direct minimization of Eq. (5.2). The solution represents the conditional mean over both the source and the channel statistics, and it is given in the following equation for the special case of squared error distortion [24, 97]

$$C_j = \frac{\sum_{i=0}^{K-1} P_{j|i} \left(\sum_{n:Y_n \in S_i} Y_n \right)}{\sum_{i=0}^{K-1} \sum_{n:Y_n \in S_i} \pi_n(i) P_{j|i}} \qquad (5.4)$$

In this equation the term $\sum_{n:Y_n \in S_i} \pi_n(i)$ specifies the source output vectors, which are allowed to contribute to the update of codeword C_i. In order to clearify the presentation of the COVQ procedure, we define the following decoding probability variable

$$\varphi_n(j) = \sum_{i=0}^{K-1} \pi_n(i) P_{j|i} \qquad (5.5)$$

where $\pi_n(i)$ is the encoding probability defined in Eq. (5.1). Note that the definition of the encoding probability takes care of the conditional subscript of $\sum_{n:Y_n \in S_i}$. Similar notation is adopted in [97]. Using Eq. (5.5) in Eq. (5.4) produces

$$C_j = \frac{\sum_n \varphi_n(j) Y_n}{\sum_n \varphi_n(j)} \qquad j = 0, 1, \ldots, K-1 \qquad (5.6)$$

Then, using an initial guess, a training sequence, and the NNC of Eq. (5.3), the LBG algorithm is employed to iteratively solve for a locally optimal COVQ codebook. The codewords are updated using the centroid equations shown in Eq. (5.6).

5.1.2 Channel Optimized Trellis Vector Quantization

The first heuristic statement of the principles underlying channel-optimized trellis vector quantization (CO-TVQ) was developed in [21], and was later implemented in [22]. In order to explain the concept of CO-TVQ, consider a TVQ system that delivers at time n the symbol x_n, which represents a branch index out of state s_n. This symbol is received at the FSM decoder input as $\hat{x}_n$. In the case of passing information over a noiseless channel, $\hat{x}_n$ along with the current FSM state $\hat{s}_n$ reproduce the exact codeword that colours the corresponding encoding trellis branch. On the other hand, this statement is never valid when operating over noisy channels, since $\hat{x}_n$ and $\hat{s}_n$ no longer match x_n and s_n. Nonetheless, from a probabilistic point of view, and given a trellis state s_n and branch index x_n, the reproduction of a particular codeword C is controlled by a probability mass function, which is given by [21]

$$\Pr\{C|s_n, x_n\} = \sum_{\hat{x},\hat{s}:\Phi(\hat{s},\hat{x})=C} Q(\hat{s}, \hat{x}|s_n, x_n) \tag{5.7}$$

where $Q(\cdot|\cdot)$ is the channel distribution, and $\Phi(\hat{s}, \hat{x})$ represents the FSM decoder function as defined in section 2.3.

In the case of COVQ, the effect of noise on the distortion of reproduction is considered through modifying the encoding rule, which is represented by the NNC of Eq. (5.3). Similarly, in order to consider the channel noise in the CO-TVQ encoding rule, Eq (5.7) intuitively suggests colouring the trellis branches with $\Pr\{\cdot|s_n, x_n\}$ rather than with $\Phi(s_n, x_n)$. Accordingly, the contribution of a particular branch x_n at state s_n to the path metric is evaluated using

$$\delta_{s_n}(Y_n, x_n) = \sum_{C \in \mathcal{C}} d(Y_n, C) \Pr\{C|s_n, x_n\} \tag{5.8}$$

This equation represents the basic metric, which along with a search algorithm constitute the channel-optimized TVQ encoding process over noisy channels. Note that the definition of $\Pr\{C|s_n, x_n\}$ in Eq. (5.7) is meant to represent the general situation, in which more than one FSM output point to the same reproduction codeword. However, in this thesis, the FSM is a deterministic finite-memory shift register, thus, the codewords are addressed by only one FSM output. In this case, the variable $\Pr\{C|s_n, x_n\}$ is simply the channel transition probability $P_{j|i}$, where j represents the index of the codeword C, and i is the index of the codeword that colours the branch x_n at state s_n.

The search for an optimal CO-TVQ codebook is performed by minimizing the average distortion given by Eq. (5.2). However, to fit the notation of CO-TVQ,

$\pi_n(i)$ is replaced by the TVQ encoding probability $\pi_n(i|Y_1^N)$, which is given by Eq. (4.33) and reproduced as follows

$$\pi_n(i|Y_1^N) = \begin{cases} 1, & \psi_n(i) > \psi_n(j) \\ 0, & \text{otherwise} \end{cases} \quad ; \quad i,j = 0,1,\ldots,2^{k\nu}-1 \tag{5.9}$$

Recall that $2^{k\nu}$ is the total number of trellis branches, thereby, representing the trellis codebook cardinality K. Besides that, $\psi_n(i) = \Pr\{C_i|Y_1^N\}$ is defined in Eq. (4.34). Accordingly, the channel-optimized average distortion is

$$D = \frac{1}{N_T}\sum_{n}\sum_{i=0}^{K-1}\sum_{j=0}^{K-1}\pi_n(i|Y_1^N)P_{j|i}d(Y_n,C_j) \tag{5.10}$$

The minimization process once again results into the centroid updates shown in Eq. (5.6), however, the decoding probability of Eq. (5.5) is defined in the CO-TVQ case as

$$\varphi_n(j) = \sum_{i=0}^{K-1}\pi_n(i|Y_1^N)P_{j|i} \tag{5.11}$$

Then, the codewords update is performed according to Eqs. (5.6) and (5.11) using the LBG algorithm along with the modified node-metric of Eq. (5.8) and a search algorithm.

The complexity of the distortion measure defined in Eq. (5.10) usually perplexes the search for a globally optimal minimum distortion point. Besides that, for several practical information sources and high quantization dimensions, a globally minimum distortion point is a rather hard-to-retain issue that cannot be described mathematically. This directly reflects into the significance of the search process initialization, especially with local descent algorithms like the LBG algorithm. However, the search process for a noisy channel-optimized codebook is less sensitive to the initialization phase, this was first reported in [25]. To explain this statement, consider the effect of channel errors on the quantization process for a given codebook. For small channel BER values ϵ, the channel errors will not result in dramatic changes of the location of a reproduction codeword in the encoding space, thus having a relatively local effect within the vicinity of that particular codeword. On the other hand, for higher values of ϵ, the errors introduced over the noisy channel will change the location of a particular codeword in a larger area of the encoding space. Therefore, over "very" noisy channels, the impact of having a particular initial codebook on the global convergence process is limited by the effect of channel errors. In other words, the local effect of the reproduction codewords becomes less significant as the BER over the noisy channel increases. Similar interpretation is presented in [123]. Despite this fact, we shall see in the simulations section, and for a practical range

of the channel BER ϵ, that the performance of the LBG algorithm is still dependent on the choice of initial codebooks. This dependency along with the need for "long" training sequences still preclude achieving lower-distortion channel-optimized codebook configurations. The authors in [97] provide a robust deterministic annealing based approach to solving the problems associated with the LBG algorithm in the context of COVQ. In the next section, we extend the FSTVQ approach presented in chapter 4 to fit the problem of CO-TVQ. The approach is denoted by CO-FSTVQ.

5.2 Channel-Optimized FSTVQ

The objective of this section is to extend the fuzzy relaxation approach (FSTVQ) presented in chapter 4 to fit the context of CO-TVQ. The extension is established using the argument of section 4.3, which culminated into employing the two basic principles of soft encoding and fuzzy association in the TVQ codebook search process. The FSTVQ approach was motivated by having measures and probability values that match those used in the development of Blahut algorithm. Blahut algorithm solves for the optimal distributions that achieve an optimal point on the rate distortion curve parameterized by a specific Lagrange multiplier. Inspired by this algorithm, the FSTVQ approach solves for the exact reproduction codewords, rather than their distribution, by gradually increasing the value of the Lagrange multiplier to arrive at a specific rate-distortion point using an association distribution that matches the one used in Blahut algorithm.

In summary, the soft encoding part of FSTVQ represents minimizing a soft average distortion measure rather than a hard one. The word "soft" is used to denote the process of weighting the contribution of the source output vectors to the different encoding sets. By virtue of having the STVQ algorithm, which delivers soft distortion-dependent reliability information, we managed in section 4.4 to define a soft average distortion measure similar to the theoretical distortion rate function. Minimizing this distortion measure produced the FSTVQ codeword update equations, which eventually provided initialization-independent lower-distortion codebooks using short training sequences. On the other hand, the fuzzy association part of the process is performed via gradually increasing the value of a Lagrange multiplier to control the degree of membership of a source output vector to an encoding set. This part is performed based on an instantaneous distortion measure.

In the CO-TVQ framework of reference, the channel-optimized average distortion measure is given by Eq. (5.10). However, in accordance with the development of

FSTVQ, the hard encoding probability $\pi_n(i|Y_1^N)$ is replaced by $\psi_n(i) = \Pr\{C_i|Y_1^N\}$ resulting into the soft channel-optimized average distortion

$$D = \frac{1}{N_T} \sum_n \sum_{i=0}^{K-1} \sum_{j=0}^{K-1} \psi_n(i) P_{j|i} d(Y_n, C_j) \tag{5.12}$$

where $\psi_n(i)$ is given by Eq. (4.34), which is represented by the following equation

$$\psi_n(i) = \frac{\lambda_n^i(m)}{\sum_{m=0}^{M-1} \sum_{i=0}^{I-1} \lambda_n^i(m)} \tag{5.13}$$

with $\lambda_n^i(m)$ defined in Eq. (4.15) and solved for using the STVQ algorithm. Furthermore, in harmony with Eq. (5.13), C_i represents the codeword associated with the trellis state $s_n = m$ and the branch index $x_n = i$. In order to solve for the codewords-update rule, we simply differentiate the average distortion given by Eq. (5.12) with respect to C_j. For the case of squared error distortion, this step produces

$$\frac{\partial D}{\partial C_j} = \frac{-2}{N_T} \sum_n \sum_{i=0}^{K-1} \psi_n(i) P_{j|i} (Y_n - C_j); \quad \forall j = 0, 1, \ldots, K-1 \tag{5.14}$$

Equating the derivative to zero results in the following codewords update equations

$$C_j = \frac{\sum_{i=0}^{K-1} \sum_n \psi_n(i) P_{j|i} Y_n}{\sum_{i=0}^{K-1} \sum_n \psi_n(i) P_{j|i}} \quad \forall j = 0, 1, \ldots, K-1 \tag{5.15}$$

At this stage, we redefine the decoding probability variable $\varphi_n(j)$, given by Eq. (5.11), using $\psi_n(i)$ instead of the hard probability variable $\pi_n(i|Y_1^N)$ as follows

$$\varphi_n(j) = \sum_{i=0}^{K-1} \psi_n(i) P_{j|i} \tag{5.16}$$

Applying Eq. (5.16) into Eq. (5.15) results in the following simplified codewords update equation

$$C_j = \frac{\sum_n \varphi_n(j) Y_n}{\sum_n \varphi_n(j)} \qquad j = 0, 1, \ldots, K-1 \tag{5.17}$$

Note that the difference between Eq. (5.17) and Eq. (5.6), as applied to CO-TVQ, is inherent in the definition of $\varphi_n(j)$. The value of $\varphi_n(j)$ in Eq. (5.11) represents only one probability variable; that is $P_{j|i}$, with i representing the output symbol delivered by the CO-TVQ at time n. On the other hand, the value of $\varphi_n(j)$ in Eq. (5.17) stands for a weighted sum of $P_{j|i}$ over the different values of i, as shown in Eq. (5.16). The weights are the soft values $\psi_n(i)$ defined in Eq. (5.13), which provide

the fuzzy association rule necessary to update the different encoding sets for the noisy channel case. Unlike CO-TVQ, at the early stages of CO-FSTVQ each source output vector contributes to updating the different encoding set centroids. As it is the case with FSTVQ, the fuzziness of the codebook update process is controlled by the Lagrange multiplier used in Eq. (4.21). This concludes the derivation of the CO-FSTVQ approach. The following represents a summary of the procedure of the CO-FSTVQ process:

1. For a particular channel bit-error rate ϵ, start with an initial "small" η, which is used in Eq. (4.21), and a random initial codebook.

2. Perform CO-TVQ over the noisy channel using the node-metric defined in Eq. (5.8) and the STVQ algorithm.

3. Use the soft information delivered by the STVQ algorithm and defined in Eq. (5.13) to update the codewords according to Eqs. (5.17) and (5.16).

4. Compute the soft average distortion defined in Eq. (5.12) using Eq. (5.13). This step is necessary to test the approach towards the optimal minimum, which is used by the inherent steps of the LBG algorithm. The algorithm halts the process when no decrease in average distortion is achieved at $\eta = \eta^*$. Recall that η^* is the value of η associated with the optimal performance of the STVQ algorithm for a given distortion and compression rate.

5. Increment η. If $\eta > \eta^*$, then $\eta = \eta^*$, go to step 2.

Note that the final stages of the CO-FSTVQ algorithm is controlled by step 4 in the above procedure. The termination stage occurs at $\eta = \eta^*$, in an approach identical to the LBG algorithm with CO-STVQ.

5.3 Performance Evaluation

In accordance with the development presented thus far in this chapter, we shall consider trellis quantization over a binary symmetric channel (BSC) with bit error rate (BER) ϵ. This channel models the case of binary modulation (BPSK) over a memoryless Gaussian channel with constant or time varying SNR and hard decision decoding. Therefore [124]

$$\epsilon = \frac{1}{2}\mathcal{F}^c(\sqrt{\frac{E_b}{N_o}}) \tag{5.18}$$

where E_b represents the energy per transmitted information bit, $N_o/2$ is the variance of the Gaussian channel noise. Besides that, $\mathcal{F}^c(\cdot)$ stands for the complementary error function, which is given by [124]

$$\mathcal{F}^c(x) = \frac{2}{\sqrt{\pi}}\int_x^\infty e^{-t^2}dt$$

To compute the channel transition probability variable $P_{j/i}$, which is used in the development of the theory of this chapter, note that the symbols j, i are $k\nu$-bit codeword indices. Accordingly, the variable $P_{j/i}$ of a BSC with BER ϵ is computed as follows

$$P_{j/i} = \epsilon^{h_d}(1-\epsilon)^{k\nu - h_d}$$

where h_d represents the Hamming distance between j and i.

Performance evaluation is performed using different trellis structures with first and second order Gauss-Markov sources. The fuzzy codebook search approach (CO-FSTVQ) is compared to the LBG algorithm, which is implemented using both STVQ and VTVQ. In order to show the advantage of using CO-FSTVQ, the codebook search process is performed 15 times, each time with a different initial codebook chosen at random from the training sequence. The same initial codebooks are used with the different algorithms for each particular case. Unlike the LBG algorithm, we demonstrated in chapter 4 that the FSTVQ approach is capable of descending towards lower distortion points using relatively short training sequences. Therefore, to accentuate this feature in the channel-optimized version of the FSTVQ algorithm, training is performed using sequences of length 2000 samples. Based on several experimental outcomes, this sequence length proved to provide the rough performance borderline sought for in the different cases considered in this chapter.

The methodology of our simulations involves two phases; training and testing. For each algorithm and given a particular compression rate, trellis structure (constraint length) and channel bit error rate ϵ, we employ the same training sequence to look for the "optimal" codebook 15 times using the different 15 initial codebooks. On

the other hand, the testing phase involves applying the "best" and the "worst" optimal codebooks, in the minimum distortion sense, to encoding 50-different 2000-sample test sequences over the corresponding noisy channel. The results are used to compute the average signal to quantization noise ratio as well as the corresponding 95% confidence interval (CI). The computation of the confidence interval is once again based on the argument presented in section 4.3. Recall that 1000-sample realizations were used in the process of average performance evaluation of chapter 4. In this chapter, however, we use 2000-sample realizations in order to increase the number of channel errors over the coding discrete-time window. For instance, the minimum BER ϵ used in the different simulation cases is 0.005. Therefore, transmitting 2000 source output samples over a channel with $\epsilon = 0.005$ is expected to produce a maximum of $(0.005 \times 2000Rk)$ channel errors. Note that R represents the compression rate, and k is the number of bits per a TVQ output symbol.

The testing phase itself has two parts; testing over the noisy channel, which is referred to as the channel-optimized performance, and testing over the noise-free channel. This last part is performed to measure the performance degradation of using channel-optimized codebooks over noise-free channels. To further complete the picture, we apply regular TVQ codebooks over the noisy channel, which demonstrates the performance loss due to channel noise as compared to channel-optimized codebooks. For this purpose, we apply FSTVQ to perform the codebook search process using the different 15 initial codebooks and the same 2000-sample training sequences. Moreover, the advantage of using channel-optimized codebooks is tested in channel mismatched conditions as compared to noiseless channel codebooks for several selected cases.

Before presenting the results it is important to discuss the theoretical rate-distortion limits of transmitting information over a noisy channel. As indicated in appendix A, we can solve for the exact values on the curve of the distortion rate function for a particular source. These theoretical values, set the lowest possible limits for theoretical compression over the noiseless channel. For the case of source coding over the noisy channel, McEliece in chapter 5 of [125] develops the *optimal performance theoretically attainable* (OPTA) limit for transmitting information sources over noisy channels. The OPTA limit $\bar{R}(D)$ represents the minimum compression rate possible to convey information over a noisy channel with capacity G and average fidelity of reproduction D. Clearly, the OPTA is conceptually related to the rate distortion function, in fact it is defined as

$$\bar{R}(D) = \frac{R(D)}{G} \tag{5.19}$$

Nevertheless, for comparison purposes, we are usually interested in the theoretical minimum distortion limit, which we shall denote hereafter by the *minimum distortion theoretically attainable* (MDTA) $D(\bar{R})$. Note that, the MDTA value for a particular source coding system is the value at the OPTA curve corresponding to the operational compression rate. However, on the noiseless distortion rate ($D(R)$) curve, the MDTA point represents the value of the distortion rate function at $R(D) = G\,\bar{R}(D)$; given that $\bar{R}(D)$ is replaced by the operational compression rate of interest.

This treatment of the OPTA concept intuitively shows the impact of channel noise on re-shaping the theoretical rate distortion curve of an information source. As an example, Figs. 5.1 and 5.2 show the OPTA curves for an AR(1) source with $\alpha = 0.9$ over a BSC. Note that the channel capacity G, which for the BSC is given by Eq. 3.3, is always less than or equals to 1 symbol/channel use (the channel symbol in our case is a bit). Therefore, to achieve a given average fidelity of reproduction $D(R)$, the channel noise actually increases the required theoretical minimum information rate ($R(D)$) by a factor of $1/G$, as shown in Fig. 5.2. This factor accommodates for the extra theoretical penalty required to balance the compression/error-protection trade-off over the noisy channel. The increased level of redundancy at the channel input is necessary to reduce the sensitivity of the transmitted sequence to channel errors. Thereby, achieving the same average distortion D that is associated with R(D) for the case of transmitting over the noiseless channel ($G = 1$).

5.3.1 AR(1) Source

In this section performance testing is conducted using the highly correlated AR(1) source with a coefficient $\rho = 0.9$. Two compression cases over the noisy channel are considered; 1 bps and 0.5 bps. The first case represents 2-dimensional compression ($L = 2$) at a rate of 1 bps using a trellis with $k = 2$ and constraint lengths $\nu = 2, 3$. Note that in order to enhance the performance of the trellis we use two-dimensional compression with $k = 2$ to achieve a rate of 1 bps. It is worth mentioning that although increasing the trellis constraint length increases the number of reproduction codewords, it is the parameter k that mainly controls partitioning the encoding space into 2^k main subsets. Aside from the fact that the former argument is supported by the physical compression rate (k/L), it was experimentally demonstrated that the training algorithm equally divides (clusters) the different codewords in the vicinity of 2^k centroids in the encoding space.

Searching for the channel-optimized codebooks is performed over the following set of channel BER

$$\underline{\epsilon} = [0.005 \;\; 0.01 \;\; 0.03 \;\; 0.05 \;\; 0.1]$$

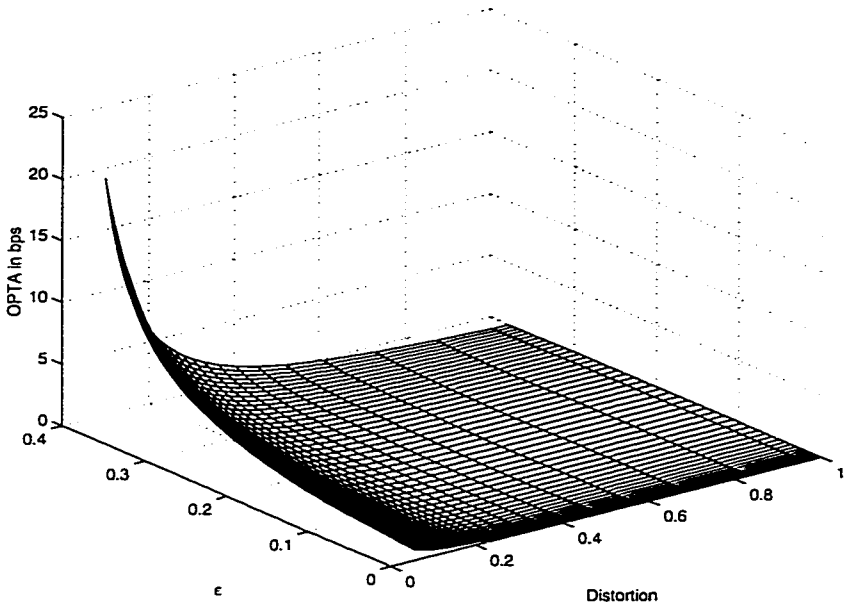

Figure 5.1: The OPTA curve for an AR(1) source with $\alpha = 0.9$ as a function of distortion and channel BER ϵ.

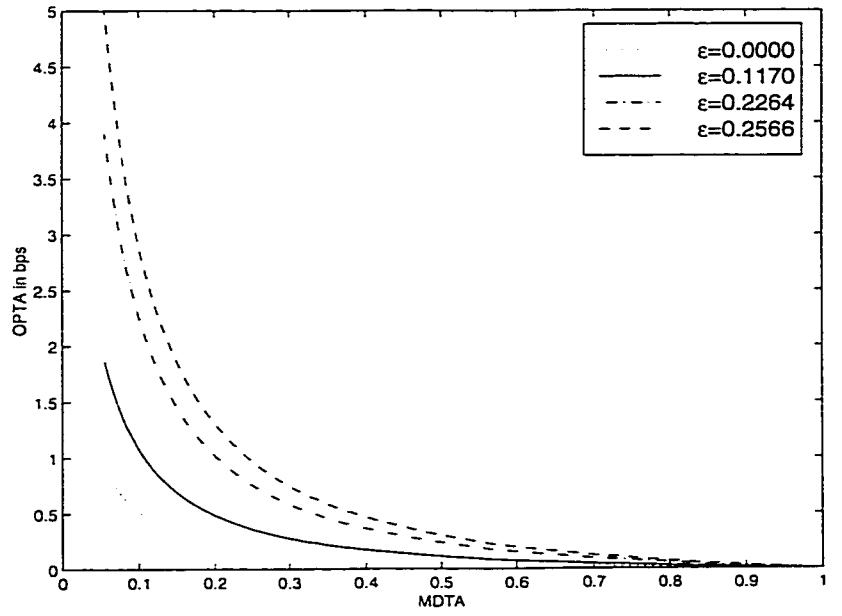

Figure 5.2: The OPTA curve for an AR(1) source with $\alpha = 0.9$ as a function of distortion.

Tables 5.1 and 5.2 show the average SNR as well as the 95% confidence interval over the corresponding noisy channel and the noiseless channel. In these tables we show SNR_{max} and SNR_{min}, which amongst the 15 independent training runs correspond to the best and worst codebooks, respectively. The **bold** numbers represent coding over channel-matched conditions, while the second-line numbers of each cell represent the performance over the error-free channel. For this particular source and compression rate, the channel-optimized TVQ (CO-TVQ) outperformed the regular source-matched TVQ (SM-TVQ), designed using FSTVQ, under noisy channel-matched conditions as shown in Table 5.1. As expected, the coding gain of the channel-optimized approach increases as the channel conditions worsen.

On the other hand, Table 5.2 shows the channel-optimized performance of the LBG-codebooks using both STVQ and VTVQ. The sensitivity of the LBG algorithm to initial codebooks and to the 2000-sample training sequence length is clear from the gap in performance between SNR_{max} and SNR_{min}. This gap is significantly reduced as the channel became more noisy. This anticipated behaviour was physically explained in the problem statement in section 5.1.2. Nonetheless, the difference in performance, as compared to the performance of CO-FSTVQ codebooks, is still at least ~ 1 dB at $\epsilon = 0.05$ and $\nu = 3$. This channel BER value is considered higher that most practical BER values, which justifies using CO-FSTVQ for the design of the trellis codebook below for instance $\epsilon = 0.05$. Furthermore, lower distortions of reproduction were achieved by CO-FSTVQ as compared to the best performance of the LBG algorithm amongst the 15 trials for low channel BER. In Figs. 5.3 and 5.4 we demonstrate the behaviour of the CO-LBG-based algorithms compared to that of the CO-FSTVQ algorithm. The impact of channel noise on reducing the dependency of the different algorithms on initial conditions is clear from Fig. 5.4(a). However, the advantage of CO-FSTVQ is clear for more complex trellis structures, Fig. 5.4(b), especially at lower channel BER as shown in Fig. 5.3.

One disadvantage of the channel-optimized performance is the degradation involved during operating under channel-mismatched conditions. As shown in Tables 5.1 and 5.2, there is a performance loss that results from applying the channel-optimized codebooks to compression over the noiseless channel as compared to the performance of SM-TVQ codebooks. The performance degradation becomes more pronounced as we employ codebooks that are matched to noisy channels with higher BER. Nevertheless, as shown in Table 5.1, there is also a certain level of performance degradation that results from applying SM-TVQ codebooks over the noisy channel. For example, in reference to the results documented in Tables 5.1 and 5.2 for $\epsilon = 0.05$ and $\nu = 3$, using the CO-TVQ over the noisy channel provides a gain of ~ 1.2 dB, while risking a loss of only ~ 0.5 dB in the event of error-free transmission. Further treatment of operating through channel mismatched conditions is considered later

Table 5.1: The CO-FSTVQ and FSTVQ rate 1 bps coding performance for the AR(1) source with $\rho = 0.9$; $k = 2$, $L = 2$. The values are listed as SNR $\pm$ CI in (dB). The **bold** numbers represent the channel optimized performance, while the other numbers represent testing over the noiseless channel.

		CO-FSTVQ		FSTVQ	
ϵ	ν	SNR_{max}	SNR_{min}	SNR_{max}	SNR_{min}
0.005	2	**9.305 ± 0.076**	**9.233 ± 0.090**	**9.184 ± 0.116**	**9.138 ± 0.116**
		9.856 ± 0.063	9.818 ± 0.066	10.300 ± 0.033	10.273 ± 0.034
	3	**10.134 ± 0.034**	**10.110 ± 0.037**	**9.967 ± 0.041**	**9.801 ± 0.064**
		10.857 ± 0.020	10.830 ± 0.023	10.869 ± 0.013	10.794 ± 0.013
0.01	2	**8.866 ± 0.087**	**8.834 ± 0.077**	**8.235 ± 0.152**	**8.208 ± 0.149**
		9.913 ± 0.063	9.908 ± 0.063	10.300 ± 0.033	10.273 ± 0.034
	3	**9.579 ± 0.058**	**9.553 ± 0.060**	**9.270 ± 0.0747**	**8.995 ± 0.066**
		10.694 ± 0.036	10.653 ± 0.039	10.869 ± 0.013	10.794 ± 0.013
0.03	2	**7.243 ± 0.135**	**7.176 ± 0.148**	**6.039 ± 0.258**	**5.995 ± 0.255**
		9.294 ± 0.108	9.188 ± 0.129	10.300 ± 0.033	10.273 ± 0.034
	3	**8.073 ± 0.091**	**8.073 ± 0.091**	**7.244 ± 0.254**	**6.941 ± 0.194**
		10.253 ± 0.063	10.253 ± 0.063	10.869 ± 0.013	10.794 ± 0.013
0.05	2	**6.364 ± 0.161**	**6.342 ± 0.163**	**4.586 ± 0.275**	**4.556 ± 0.278**
		8.520 ± 0.157	8.489 ± 0.160	10.300 ± 0.033	10.273 ± 0.034
	3	**7.553 ± 0.098**	**7.553 ± 0.098**	**6.307 ± 0.115**	**5.903 ± 0.122**
		10.471 ± 0.061	10.471 ± 0.061	10.869 ± 0.013	10.794 ± 0.013
0.1	2	**4.987 ± 0.214**	**4.987 ± 0.214**	**2.347 ± 0.387**	**2.345 ± 0.393**
		7.517 ± 0.220	7.517 ± 0.220	10.300 ± 0.033	10.273 ± 0.034
	3	**5.719 ± 0.173**	**5.719 ± 0.173**	**4.522 ± 0.174**	**3.969 ± 0.242**
		8.549 ± 0.173	8.549 ± 0.173	10.869 ± 0.013	10.794 ± 0.013

Table 5.2: The CO-STVQ and CO-VTVQ rate 1 bps coding performance for the AR(1) source with $\rho = 0.9$; $k = 2$, $L = 2$. The values are listed as SNR $\pm$ CI in (dB). The **bold** numbers represent the channel optimized performance, while the other numbers represent testing over the noiseless channel.

		CO-STVQ		CO-VTVQ	
ϵ	ν	SNR_{max}	SNR_{min}	SNR_{max}	SNR_{min}
0.005	2	**9.379 ± 0.066**	**7.406 ± 0.179**	**9.381 ± 0.068**	**7.414 ± 0.156**
		9.981 ± 0.054	8.619 ± 0.091	9.984 ± 0.049	8.622 ± 0.097
	3	**9.780 ± 0.063**	**7.931 ± 0.252**	**9.348 ± 0.079**	**7.986 ± 0.256**
		10.500 ± 0.035	9.300 ± 0.077	10.240 ± 0.038	9.350 ± 0.079
0.01	2	**8.887 ± 0.076**	**7.458 ± 0.192**	**8.846 ± 0.074**	**7.424 ± 0.184**
		9.966 ± 0.057	9.009 ± 0.086	9.976 ± 0.052	9.037 ± 0.086
	3	**9.414 ± 0.076**	**7.919 ± 0.221**	**9.411 ± 0.090**	**7.787 ± 0.143**
		10.526 ± 0.044	9.840 ± 0.069	10.605 ± 0.047	9.576 ± 0.073
0.03	2	**7.252 ± 0.153**	**6.371 ± 0.252**	**7.256 ± 0.150**	**6.381 ± 0.248**
		9.306 ± 0.108	8.679 ± 0.137	9.300 ± 0.108	8.671 ± 0.138
	3	**8.436 ± 0.110**	**6.679 ± 0.280**	**8.464 ± 0.073**	**6.780 ± 0.277**
		10.477 ± 0.046	10.095 ± 0.059	10.564 ± 0.045	10.185 ± 0.051
0.05	2	**6.342 ± 0.143**	**6.275 ± 0.187**	**6.338 ± 0.149**	**6.285 ± 0.174**
		9.317 ± 0.107	9.444 ± 0.086	9.288 ± 0.114	9.498 ± 0.082
	3	**7.545 ± 0.103**	**6.525 ± 0.134**	**7.548 ± 0.101**	**6.626 ± 0.114**
		10.398 ± 0.060	9.417 ± 0.106	10.379 ± 0.062	9.368 ± 0.121
0.1	2	**4.996 ± 0.181**	**5.003 ± 0.209**	**4.997 ± 0.181**	**5.005 ± 0.207**
		7.519 ± 0.219	7.568 ± 0.218	7.525 ± 0.219	7.583 ± 0.217
	3	**5.753 ± 0.168**	**5.666 ± 0.237**	**5.743 ± 0.171**	**5.625 ± 0.241**
		8.667 ± 0.168	9.752 ± 0.081	8.647 ± 0.170	9.739 ± 0.075

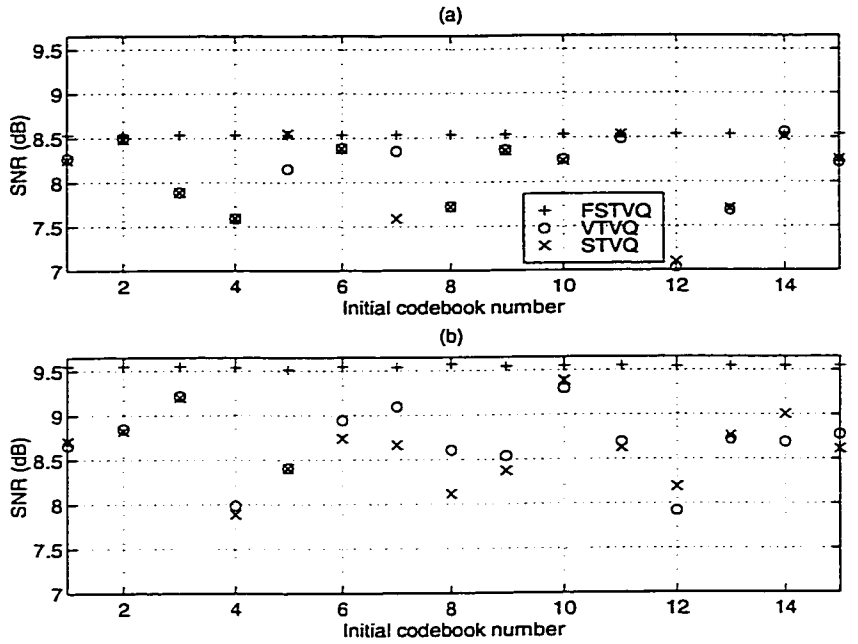

Figure 5.3: The channel-optimized coding performance of the AR(1) source at a rate of 1 bps, $k = 2, L = 2$, and $\epsilon = 0.01$, using 2000 training samples: (a) $\nu = 2$, (b) $\nu = 3$.

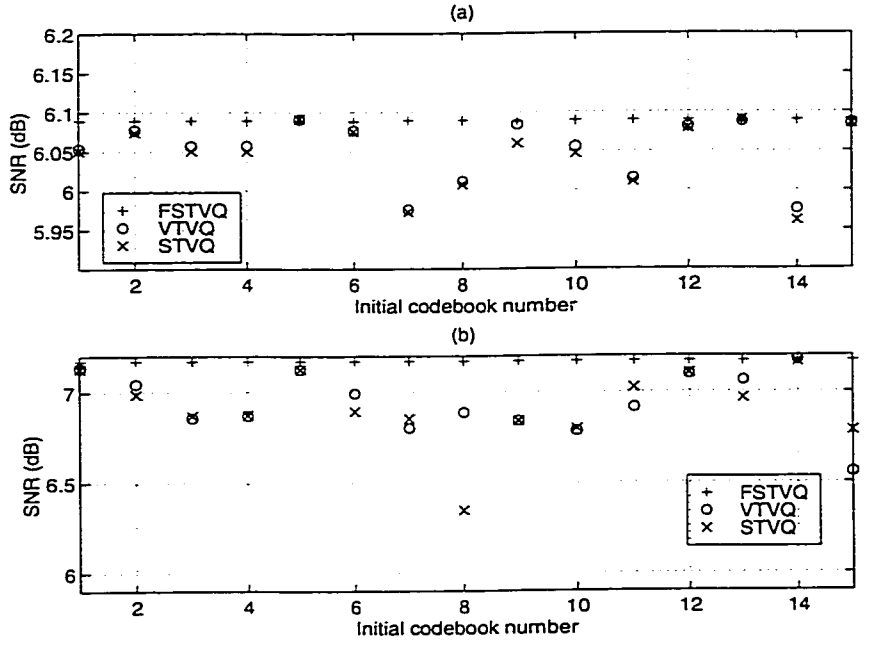

Figure 5.4: The channel-optimized coding performance of the AR(1) source at a rate of 1 bps, $k = 2, L = 2$, and $\epsilon = 0.05$, using 2000 training samples: (a) $\nu = 2$, (b) $\nu = 3$.

in section 5.3.3.

In the second case study, we perform 0.5 bps compression using 4-dimensional vectors and a trellis with $k = 2$ and $\nu = 2, 3$. The results are shown in Tables 5.3 and 5.4. As demonstrated in the previous 1 bps case, similar algorithms behaviour is expected in this case. However, due to the increased coding dimension the advantage of using the CO-FSTVQ approach is more evident. Once again as the performance gap between SNR_{max} and SNR_{min} of the CO-LBG algorithm increased with reduced levels of noise, CO-FSTVQ showed less sensitivity to initializations at this relatively short training sequence length. Furthermore, CO-FSTVQ provided lower average distortion of reproductions especially for $\epsilon < 0.03$ as compared to the best performance provided by the CO-LBG algorithm in Table 5.4. The relatively poor performance of the CO-LBG algorithm is demonstrated in Figs. 5.5 and 5.6 for the special cases of $\epsilon = 0.01, 0.05$, respectively, as compared to the performance of CO-FSTVQ. As a comparison point between the performance of the CO-TVQ and the SM-TVQ, consider in Table 5.3 the results that correspond to $\nu = 3$ and $\epsilon = 0.03$. The performance loss due to using the CO-FSTVQ codebook over the error-free channel is ~ 0.3dB, however, the same codebook provides a gain of ~ 0.65 dB over the matched noisy channel.

Table 5.3: The CO-FSTVQ and FSTVQ rate 0.5 bps coding performance for the AR(1) source with $\rho = 0.9$; $k = 2$, $L = 4$. The values are listed as SNR $\pm$ CI in (dB). The **bold** numbers represent the channel optimized performance, while the other numbers represent testing over the noiseless channel.

		CO-FSTVQ		FSTVQ	
ϵ	ν	SNR_{max}	SNR_{min}	SNR_{max}	SNR_{min}
0.005	2	**7.738 ± 0.052**	**7.687 ± 0.0527**	**7.565 ± 0.056**	**7.523 ± 0.065**
		8.074 ± 0.037	7.930 ± 0.047	8.128 ± 0.015	8.113 ± 0.017
	3	**7.900 ± 0.035**	**7.721 ± 0.058**	**7.835 ± 0.060**	**7.752 ± 0.035**
		8.301 ± 0.018	8.127 ± 0.018	8.311 ± 0.013	8.180 ± 0.020
0.01	2	**7.222 ± 0.087**	**7.095 ± 0.101**	**6.812 ± 0.136**	**6.764 ± 0.148**
		7.976 ± 0.031	7.835 ± 0.095	8.128 ± 0.015	8.113 ± 0.017
	3	**7.534 ± 0.0667**	**7.399 ± 0.112**	**7.307 ± 0.118**	**7.195 ± 0.101**
		8.338 ± 0.020	8.207 ± 0.029	8.311 ± 0.013	8.180 ± 0.020
0.03	2	**6.054 ± 0.151**	**6.027 ± 0.155**	**5.162 ± 0.171**	**5.227 ± 0.174**
		7.355 ± 0.113	7.323 ± 0.114	8.128 ± 0.015	8.113 ± 0.017
	3	**6.512 ± 0.115**	**6.502 ± 0.104**	**5.853 ± 0.140**	**5.877 ± 0.118**
		7.995 ± 0.058	7.939 ± 0.0648	8.311 ± 0.013	8.180 ± 0.020
0.05	2	**5.335 ± 0.178**	**5.317 ± 0.171**	**3.922 ± 0.224**	**3.864 ± 0.257**
		7.089 ± 0.137	7.040 ± 0.138	8.128 ± 0.015	8.113 ± 0.017
	3	**5.793 ± 0.117**	**5.779 ± 0.119**	**4.860 ± 0.129**	**4.778 ± 0.192**
		7.919 ± 0.0896	7.929 ± 0.087	8.311 ± 0.013	8.180 ± 0.020
0.1	2	**4.220 ± 0.248**	**4.220 ± 0.248**	**1.921 ± 0.326**	**1.843 ± 0.385**
		6.514 ± 0.187	6.514 ± 0.187	8.128 ± 0.015	8.113 ± 0.017
	3	**4.711 ± 0.205**	**4.711 ± 0.205**	**3.314 ± 0.138**	**3.240 ± 0.279**
		7.162 ± 0.140	7.162 ± 0.140	8.311 ± 0.013	8.180 ± 0.020

Table 5.4: The CO-STVQ and CO-VTVQ rate 0.5 bps coding performance for the AR(1) source with $\rho = 0.9$; $k = 2$, $L = 4$. The values are listed as SNR $\pm$ CI in (dB). The **bold** numbers represent the channel optimized performance, while the other numbers represent testing over the noiseless channel.

		CO-STVQ		CO-VTVQ	
ϵ	ν	SNR_{max}	SNR_{min}	SNR_{max}	SNR_{min}
0.005	2	**7.326 ± 0.076**	**6.111 ± 0.143**	**7.153 ± 0.086**	**6.086 ± 0.141**
		7.597 ± 0.068	6.697 ± 0.093	7.489 ± 0.074	6.649 ± 0.099
	3	**6.807 ± 0.110**	**5.990 ± 0.142**	**6.663 ± 0.127**	**6.100 ± 0.151**
		7.319 ± 0.068	6.551 ± 0.116	7.313 ± 0.064	6.723 ± 0.107
0.01	2	**7.004 ± 0.144**	**5.423 ± 0.342**	**6.791 ± 0.166**	**5.406 ± 0.376**
		7.67 ± 0.071	6.642 ± 0.101	7.482 ± 0.088	6.622 ± 0.100
	3	**6.307 ± 0.232**	**5.483 ± 0.341**	**6.597 ± 0.150**	**5.500 ± 0.345**
		7.411 ± 0.072	6.823 ± 0.108	7.663 ± 0.032	6.871 ± 0.099
0.03	2	**6.046 ± 0.102**	**4.939 ± 0.287**	**6.107 ± 0.103**	**5.171 ± 0.114**
		7.316 ± 0.103	6.937 ± 0.121	7.399 ± 0.097	7.321 ± 0.057
	3	**6.467 ± 0.052**	**4.707 ± 0.239**	**6.491 ± 0.052**	**4.672 ± 0.244**
		7.953 ± 0.036	6.938 ± 0.140	8.020 ± 0.037	6.905 ± 0.146
0.05	2	**5.328 ± 0.176**	**5.212 ± 0.196**	**5.327 ± 0.176**	**5.162 ± 0.240**
		7.063 ± 0.140	7.045 ± 0.127	7.08 ± 0.137	7.422 ± 0.088
	3	**5.759 ± 0.179**	**5.356 ± 0.159**	**5.750 ± 0.145**	**5.206 ± 0.176**
		7.923 ± 0.056	7.362 ± 0.108	7.797 ± 0.061	7.363 ± 0.102
0.1	2	**4.278 ± 0.228**	**3.921 ± 0.423**	**4.276 ± 0.227**	**3.914 ± 0.426**
		6.421 ± 0.184	7.165 ± 0.111	6.451 ± 0.183	7.157 ± 0.111
	3	**4.625 ± 0.182**	**4.590 ± 0.229**	**4.635 ± 0.178**	**4.516 ± 0.322**
		7.227 ± 0.133	7.123 ± 0.128	7.211 ± 0.135	7.667 ± 0.070

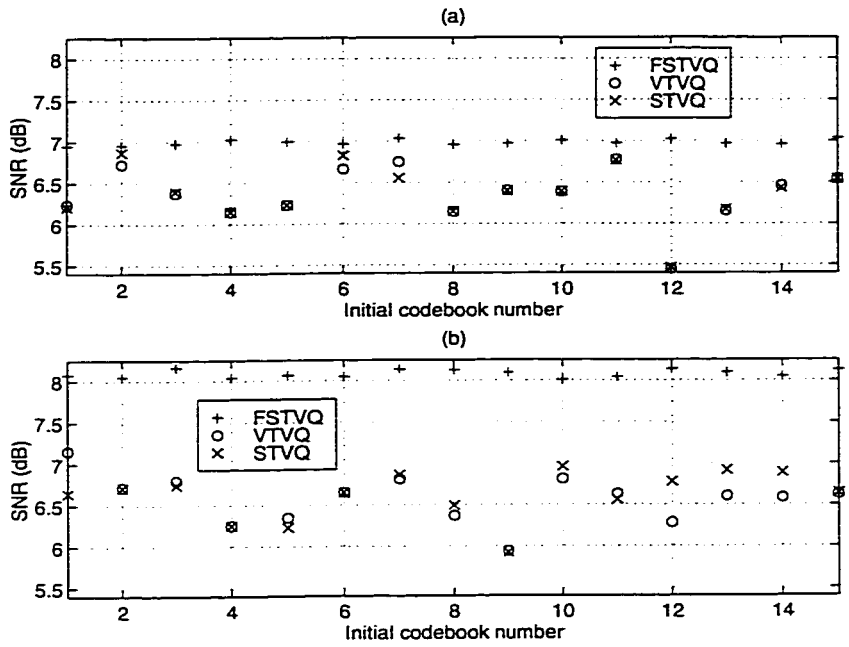

Figure 5.5: The channel-optimized coding performance of the AR(1) source at a rate of 0.5 bps, $k = 2, L = 4$, and $\epsilon = 0.01$, using 2000 training samples: (a) $\nu = 2$, (b) $\nu = 3$.

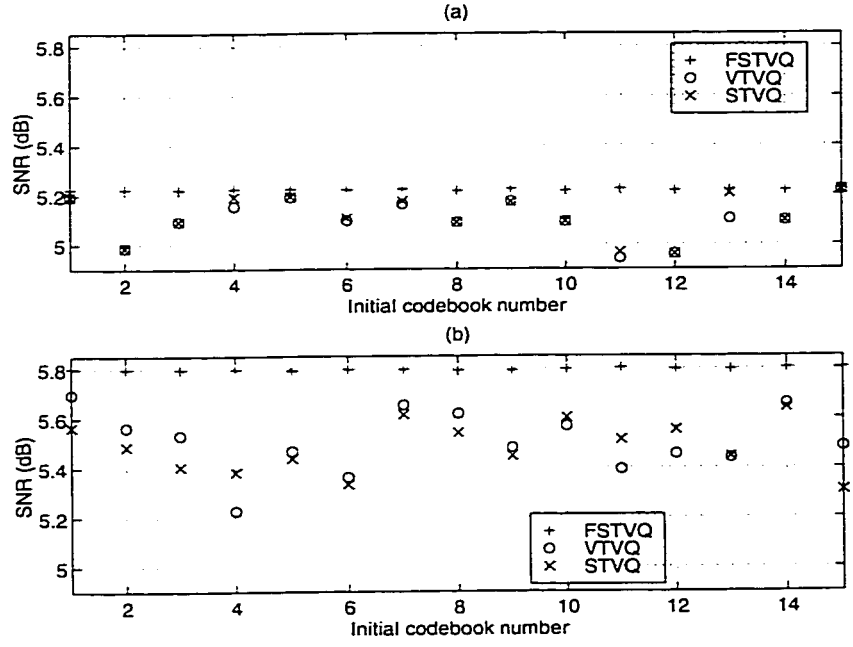

Figure 5.6: The channel-optimized coding performance of the AR(1) source at a rate of 0.5 bps, $k = 2, L = 4$, and $\epsilon = 0.05$, using 2000 training samples: (a) $\nu = 2$, (b) $\nu = 3$.

5.3.2 AR(2) Source

The AR(2) source used in this section is the one used in chapter 4. The source has the two coefficients $a_1 = 1.515, a_2 = -0.752$, which are computed to match a sample speech signal [14]. Two-dimensional vector compression is performed using a trellis with $k = 1$, $\nu = 3, 4, 5$, hence, and encoding rate of 0.5 bps. The codebook optimization is performed over the same range of channel BER used in the previous section. The results of using the different algorithms are presented in Tables 5.5 and 5.6, where we show the best and the worst performance amongst the 15 training experiments. It is interesting to demonstrate that the performance of the FSTVQ approach, for this source and at this particular compression rate, is completely independent of the initialization process. To explain this point, consider the power spectral density of this source, which is given in Eq. (A.20) and plotted in Fig. 5.7 along with the power spectral density of the AR(1) source with $\rho = 0.9$. It is clear that the spectrum of the AR(2) source covers a wider range of frequencies as compared to the AR(1) source. Therefore, it is more likely for an initial two-dimensional codebook, which is chosen at random from a sample AR(2) sequence, to represent a wider class of samples. On the other hand, due to the selectivity of the power spectral density of the AR(1) source, some of the initial codewords are expected to represent smaller number of training vectors. This explains the different sensitivity levels the FSTVQ approach exhibited with the two AR(1) and AR(2) sources.

The spectral richness of this source did not hinder the sensitivity of the CO-LBG algorithm to the initialization phase for $\epsilon \leq 0.05$. This is clear from the gap between SNR_{max} and SNR_{min} as shown in Table 5.6. Although the CO-LBG algorithm provided slightly higher SNR_{max} at $\epsilon = 0.05$ in Table 5.6 compared to the CO-FSTVQ performance in Table 5.6, the two performances are within the same 95% confidence interval. Furthermore, the CO-FSTVQ provided a steady low distortion of reproduction for the entire range of channel BER. Note that as expected the sensitivity of the LBG algorithm is significantly reduced over a very noisy channel with $\epsilon = 0.1$. As for the comparison between the SM-TVQ, which is designed by FSTVQ, and the CO-TVQ approach, we can demonstrate the increased performance loss of the SM-TVQ codebook as the channel becomes more noisy. As an example we consider the case in Table 5.5 with $\epsilon = 0.05$ and $\nu = 6$. For this case we demonstrate the importance of employing a CO-TVQ codebook over the matched channel, which provides a compression gain of ~ 1.3 dB, compared to a gain loss of only ~ 0.27 dB over the error-free channel.

Figs. 5.8 and 5.9 show the fluctuation in performance of the CO-LBG-based algorithms as compared to the consistent performance of the CO-FSTVQ. In Fig.

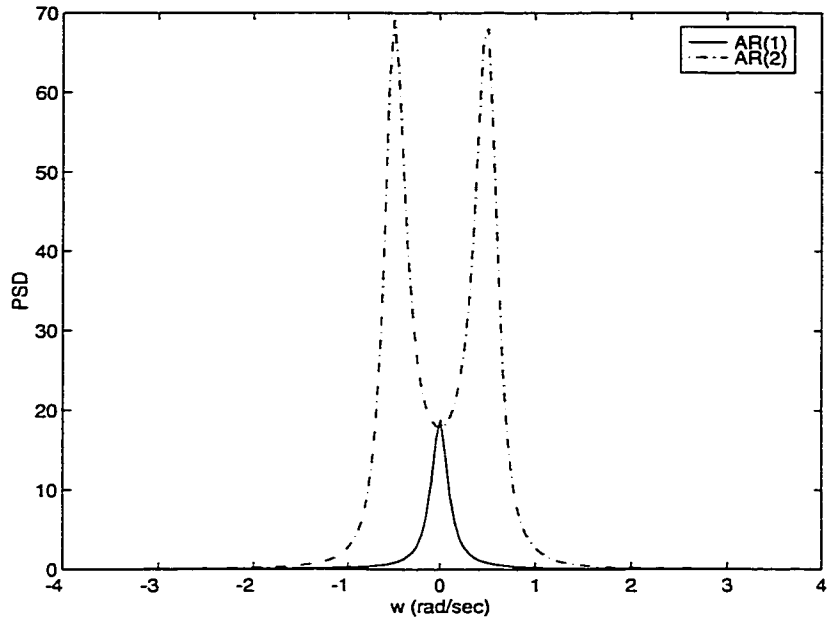

Figure 5.7: The power spectral density of the AR(1) source and the AR(2) source

5.8 two cases of optimizing over a channel with $\epsilon = 0.01$ is demonstrated, while Fig. 5.9 represents the results of optimizing over a channel with $\epsilon = 0.05$. Note the reduced sensitivity of the CO-LBG algorithm to initialization for increased levels of channel noise represented by Fig 5.9.

Table 5.5: The CO-FSTVQ and FSTVQ rate 0.5 bps coding performance for the AR(2) source; $k = 1$, $L = 2$. The values are listed as SNR $\pm$ CI in (dB). The **bold** numbers represent the channel optimized performance, while the other numbers represent testing over the noiseless channel.

		CO-FSTVQ		FSTVQ	
ϵ	ν	SNR_{max}	SNR_{min}	SNR_{max}	SNR_{min}
0.005	4	**6.567 ± 0.136**	**6.567 ± 0.136**	**6.707 ± 0.115**	**6.707 ± 0.115**
		6.813 ± 0.115	6.813 ± 0.115	7.021 ± 0.092	7.021 ± 0.092
	5	**7.012 ± 0.106**	**7.012 ± 0.106**	**6.854 ± 0.099**	**6.854 ± 0.099**
		7.195 ± 0.089	7.195 ± 0.089	7.232 ± 0.082	7.232 ± 0.082
	6	**7.112 ± 0.089**	**7.112 ± 0.089**	**6.992 ± 0.123**	**6.992 ± 0.123**
		7.462 ± 0.062	7.462 ± 0.062	7.526 ± 0.043	7.526 ± 0.043
0.01	4	**6.390 ± 0.130**	**6.390 ± 0.130**	**6.337 ± 0.124**	**6.337 ± 0.124**
		6.900 ± 0.121	6.900 ± 0.121	7.021 ± 0.092	7.021 ± 0.092
	5	**6.688 ± 0.129**	**6.688 ± 0.129**	**6.448 ± 0.157**	**6.448 ± 0.157**
		7.290 ± 0.094	7.290 ± 0.094	7.232 ± 0.082	7.232 ± 0.082
	6	**6.839 ± 0.116**	**6.839 ± 0.116**	**6.402 ± 0.197**	**6.402 ± 0.197**
		7.552 ± 0.061	7.552 ± 0.061	7.526 ± 0.043	7.526 ± 0.043
0.03	4	**5.491 ± 0.198**	**5.491 ± 0.198**	**5.223 ± 0.188**	**5.223 ± 0.188**
		6.820 ± 0.135	6.820 ± 0.135	7.021 ± 0.092	7.021 ± 0.092
	5	**5.847 ± 0.180**	**5.847 ± 0.180**	**5.314 ± 0.150**	**5.314 ± 0.150**
		7.332 ± 0.114	7.332 ± 0.114	7.232 ± 0.082	7.232 ± 0.082
	6	**5.949 ± 0.176**	**5.949 ± 0.176**	**4.774 ± 0.211**	**4.774 ± 0.211**
		7.53 ± 0.107	7.53 ± 0.107	7.526 ± 0.043	7.526 ± 0.043
0.05	4	**4.796 ± 0.266**	**4.796 ± 0.266**	**4.373 ± 0.189**	**4.373 ± 0.189**
		6.660 ± 0.157	6.660 ± 0.157	7.021 ± 0.092	7.021 ± 0.092
	5	**5.069 ± 0.221**	**5.069 ± 0.221**	**4.361 ± 0.243**	**4.361 ± 0.243**
		7.102 ± 0.132	7.102 ± 0.132	7.232 ± 0.082	7.232 ± 0.082
	6	**5.096 ± 0.212**	**5.096 ± 0.212**	**3.710 ± 0.342**	**3.710 ± 0.342**
		7.260 ± 0.136	7.260 ± 0.136	7.526 ± 0.043	7.526 ± 0.043
0.1	4	**3.556 ± 0.337**	**3.556 ± 0.337**	**2.921 ± 0.218**	**2.921 ± 0.218**
		6.015 ± 0.220	6.015 ± 0.220	7.021 ± 0.092	7.021 ± 0.092
	5	**3.847 ± 0.288**	**3.847 ± 0.288**	**2.955 ± 0.234**	**2.955 ± 0.234**
		6.412 ± 0.197	6.412 ± 0.197	7.232 ± 0.082	7.232 ± 0.082
	6	**3.950 ± 0.280**	**3.950 ± 0.280**	**2.108 ± 0.210**	**2.108 ± 0.210**
		6.625 ± 0.194	6.625 ± 0.194	7.526 ± 0.043	7.526 ± 0.043

Table 5.6: The CO-STVQ and CO-VTVQ rate 0.5 bps coding performance for the AR(2) source; $k = 1$, $L = 2$. The values are listed as SNR $\pm$ CI in (dB). The **bold** numbers represent the channel optimized performance, while the other numbers represent testing over the noiseless channel.

		CO-STVQ		CO-VTVQ	
ϵ	ν	SNR_{max}	SNR_{min}	SNR_{max}	SNR_{min}
0.005	4	**6.539 ± 0.128**	**5.384 ± 0.230**	**6.539 ± 0.128**	**5.271 ± 0.201**
		6.912 ± 0.097	5.753 ± 0.137	6.912 ± 0.097	5.623 ± 0.146
	5	**6.692 ± 0.127**	**5.213 ± 0.234**	**6.698 ± 0.141**	**5.310 ± 0.209**
		7.064 ± 0.080	5.644 ± 0.154	7.061 ± 0.086	5.758 ± 0.138
	6	**6.224 ± 0.142**	**5.583 ± 0.219**	**6.165 ± 0.185**	**5.743 ± 0.198**
		6.632 ± 0.097	6.103 ± 0.113	6.710 ± 0.097	6.311 ± 0.116
0.01	4	**6.294 ± 0.142**	**5.065 ± 0.289**	**6.167 ± 0.118**	**4.980 ± 0.118**
		6.805 ± 0.102	6.198 ± 0.148	6.759 ± 0.098	6.098 ± 0.104
	5	**6.380 ± 0.179**	**5.367 ± 0.365**	**6.379 ± 0.189**	**5.478 ± 0.269**
		7.122 ± 0.105	6.402 ± 0.144	7.119 ± 0.106	6.393 ± 0.115
	6	**6.203 ± 0.229**	**5.264 ± 0.261**	**6.314 ± 0.152**	**5.416 ± 0.325**
		7.125 ± 0.090	6.349 ± 0.123	7.167 ± 0.095	6.527 ± 0.127
0.03	4	**5.326 ± 0.205**	**4.118 ± 0.269**	**5.487 ± 0.198**	**4.192 ± 0.314**
		6.654 ± 0.143	5.863 ± 0.190	6.810 ± 0.133	6.222 ± 0.183
	5	**5.853 ± 0.184**	**4.117 ± 0.414**	**5.750 ± 0.198**	**4.424 ± 0.351**
		7.331 ± 0.112	5.989 ± 0.169	7.248 ± 0.104	6.270 ± 0.186
	6	**5.784 ± 0.184**	**4.374 ± 0.296**	**5.769 ± 0.201**	**4.490 ± 0.324**
		7.342 ± 0.117	6.382 ± 0.168	7.290 ± 0.126	6.634 ± 0.151
0.05	4	**4.804 ± 0.265**	**4.740 ± 0.254**	**4.794 ± 0.269**	**4.662 ± 0.236**
		6.696 ± 0.153	6.565 ± 0.168	6.654 ± 0.158	6.561 ± 0.168
	5	**5.111 ± 0.216**	**3.707 ± 0.401**	**5.015 ± 0.221**	**3.654 ± 0.415**
		7.193 ± 0.131	5.917 ± 0.190	7.080 ± 0.132	5.975 ± 0.197
	6	**5.104 ± 0.231**	**4.031 ± 0.331**	**5.117 ± 0.182**	**4.191 ± 0.321**
		7.255 ± 0.138	6.275 ± 0.179	7.270 ± 0.113	6.371 ± 0.179
0.1	4	**3.614 ± 0.257**	**3.574 ± 0.308**	**3.614 ± 0.256**	**3.526 ± 0.271**
		6.161 ± 0.215	6.109 ± 0.219	6.161 ± 0.214	6.023 ± 0.225
	5	**3.878 ± 0.251**	**3.803 ± 0.286**	**3.879 ± 0.261**	**3.597 ± 0.294**
		6.573 ± 0.185	6.387 ± 0.213	6.579 ± 0.181	6.183 ± 0.211
	6	**3.943 ± 0.258**	**3.804 ± 0.308**	**3.890 ± 0.269**	**3.781 ± 0.291**
		6.712 ± 0.175	6.360 ± 0.220	6.527 ± 0.187	6.364 ± 0.201

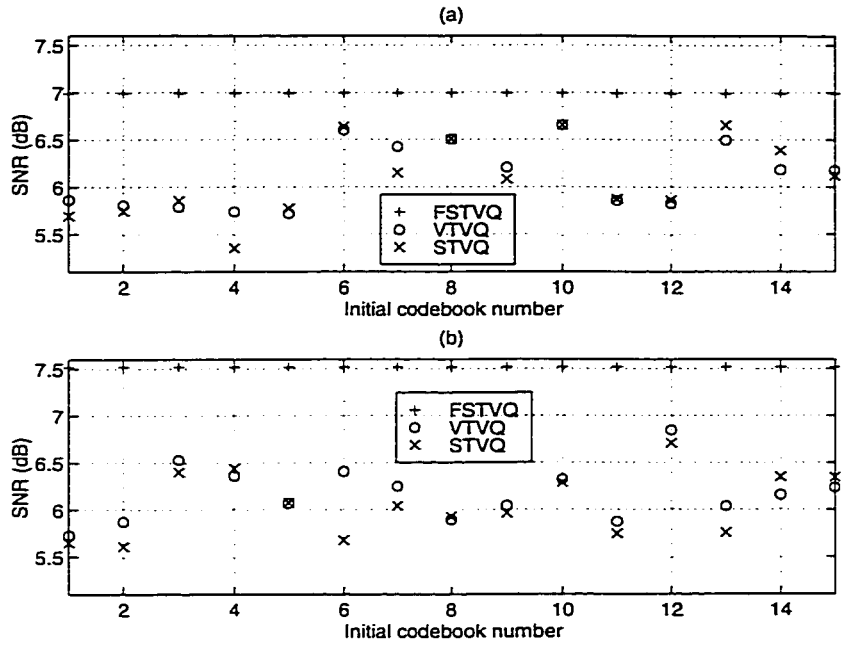

Figure 5.8: The channel-optimized coding performance of the AR(2) source at a rate of 0.5 bps, $k = 1, L = 2$, and $\epsilon = 0.01$, using 2000 training samples: (a) $\nu = 5$, (b) $\nu = 6$.

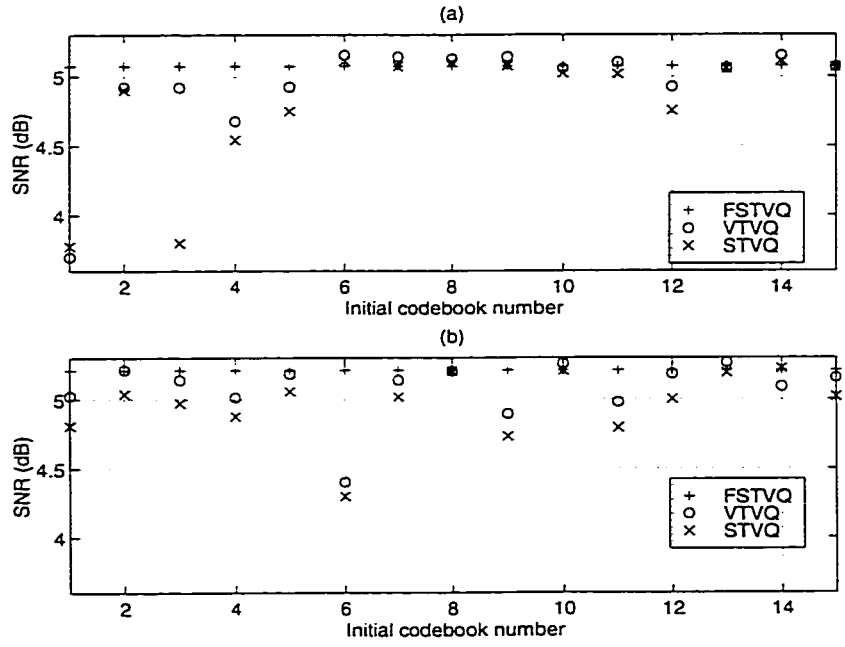

Figure 5.9: The channel-optimized coding performance of the AR(1) source at a rate of 0.5 bps, $k = 1, L = 2$, and $\epsilon = 0.05$, using 2000 training samples: (a) $\nu = 5$, (b) $\nu = 6$.

5.3.3 Channel Mismatch Conditions

In this section we investigate the effect of channel mismatched conditions on the performance of CO-TVQ codebooks as compared to the performance of SM-TVQ codebooks. This kind of study is always associated with channel optimized compression approaches, since it measures the robustness of the designed codebooks over nonstationary channels, which represent many practical communication channels [22], [24], [126] and [127]. The impact of the design BER as opposed to the actual operating values is of significant importance in the design procedure. Based on this design constraint, it is logical to minimize an average distortion value that is a function of both a design BER ϵ_d and a random actual BER ϵ_a; denoted by $D(\epsilon_d, \epsilon_a)$. Farvardin in [24] presented such a distortion function, however, due to the complexity of that function he finds a solution for the optimal ϵ_d in the range of very small values of ϵ_a ($l\epsilon_a \ll 1$), with l representing the number of bits per codeword. The solution is found to be $\epsilon_d = E\{\epsilon_a\}$. However, recall that the significance of using channel optimized quantization is for reasonably large values of channel BER. Therefore, this channel mismatched design criterion not only assumes the *a priori* knowledge of the BER distribution, but also it is practically of no value.

Instead of attempting to search for a solution to a rather complex distortion function $D(\epsilon_d, \epsilon_a)$, with an unknown ϵ_a distribution, we can just investigate the dependency of the system performance on the choice of ϵ_d. Then, in harmony with the real environment, and using some engineering sense, we can chose the proper value of ϵ_d in the light of the available tradeoffs. As an example we employed the codebooks generated using CO-FSTVQ at $\epsilon_d = 0.03, 0.05$, and the source-matched codebooks (FSTVQ) for compression over the channels with the following values of BER

$$\underline{\epsilon_a} = [0.0001 \;\; 0.001 \;\; 0.005 \;\; 0.01 \;\; 0.03 \;\; 0.05 \;\; 0.1 \;\; 0.2]$$

In Fig. 5.10 we show the results of this application using the codebooks that are associated with $\nu = 2$ in Table 5.1. Similarly, the results in Fig. 5.11 are for the codebooks that are associated with $\nu = 2$ in Table 5.3. The last case is shown in Fig. 5.12, which is the result of applying the codebooks associated with $\nu = 5$ in Table 5.5 of the AR(2) source. A common conclusion from Figs. 5.10 and 5.11 is the fact that source-matched codebooks perform better than channel-matched ones at very low channel BER. Now, this is a known fact. However, for the speech-like AR(2) source, and for this particular example the the codebook with $\epsilon_d = 0.03$ performs slightly better, although within the same CI, at very low channel BER.

Further investigation of the different curves of Figs. 5.10, 5.11 and 5.12 shows that the codebooks associated with $\epsilon_d = 0.05$ have an average performance that is slightly better than the performance of the codebooks associated with $\epsilon_d = 0.03$ for

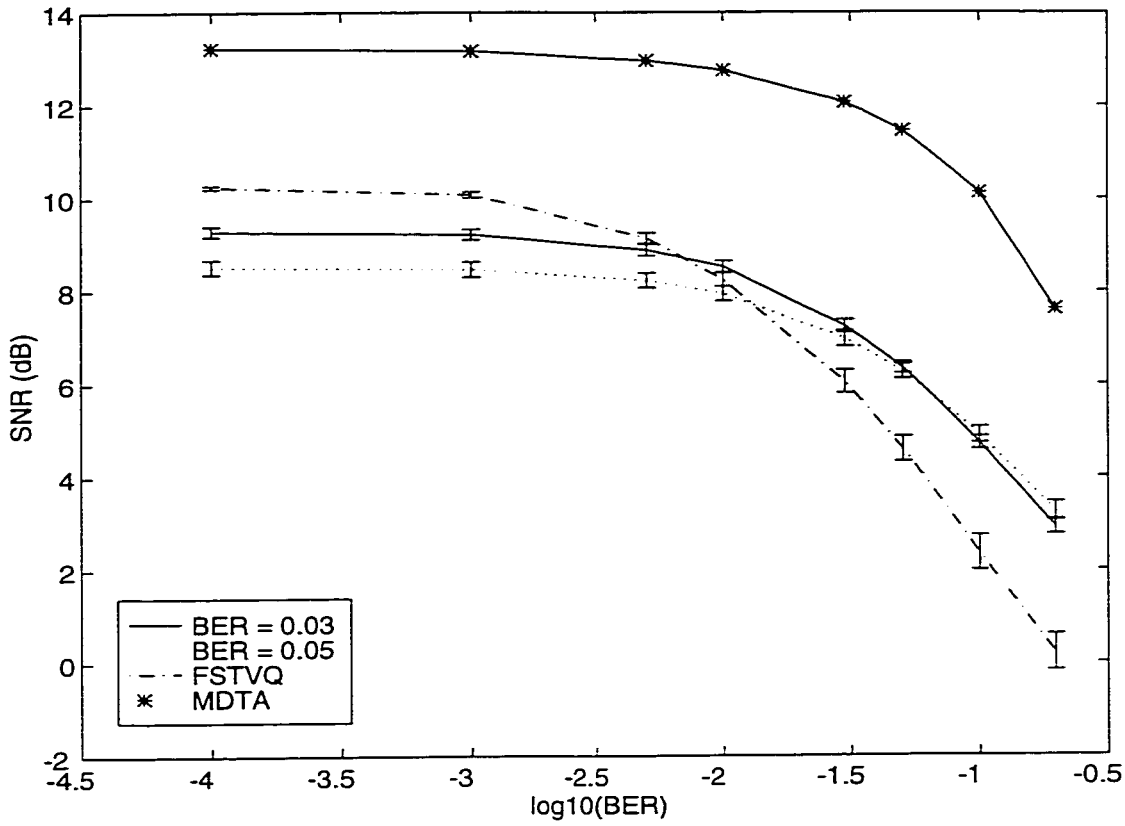

Figure 5.10: Channel mismatched performance using codebooks generated by CO-FSTVQ and FSTVQ. Two CO-FSTVQ codebooks are used; one associated with $\epsilon = 0.03$ and the other one with $\epsilon = 0.05$. The source is AR(1) and the compression rate is 1 bps with trellis parameters $k = 2$. $L = 2$ and $\nu = 2$.

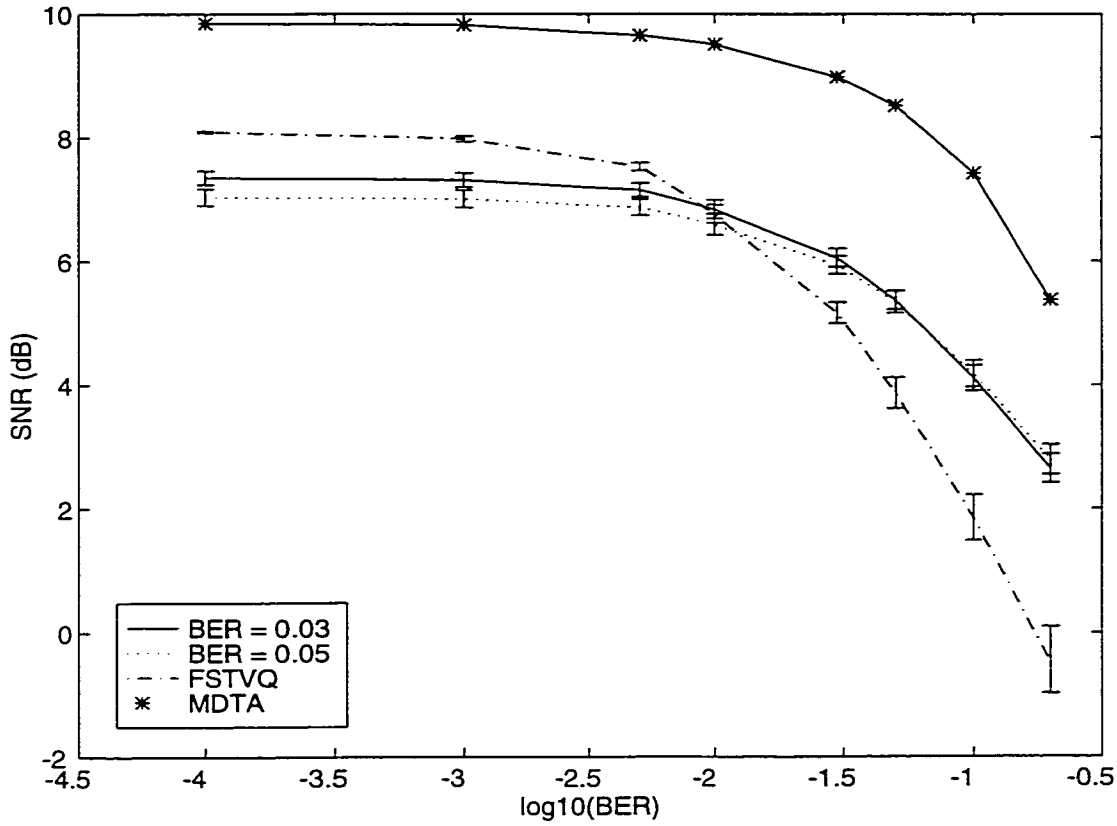

Figure 5.11: Channel mismatched performance using codebooks generated by CO-FSTVQ and FSTVQ. Two CO-FSTVQ codebooks are used; one associated with $\epsilon = 0.03$ and the other one with $\epsilon = 0.05$. The source is AR(1) and the compression rate is 0.5 bps with trellis parameters $k = 2$, $L = 4$ and $\nu = 2$.

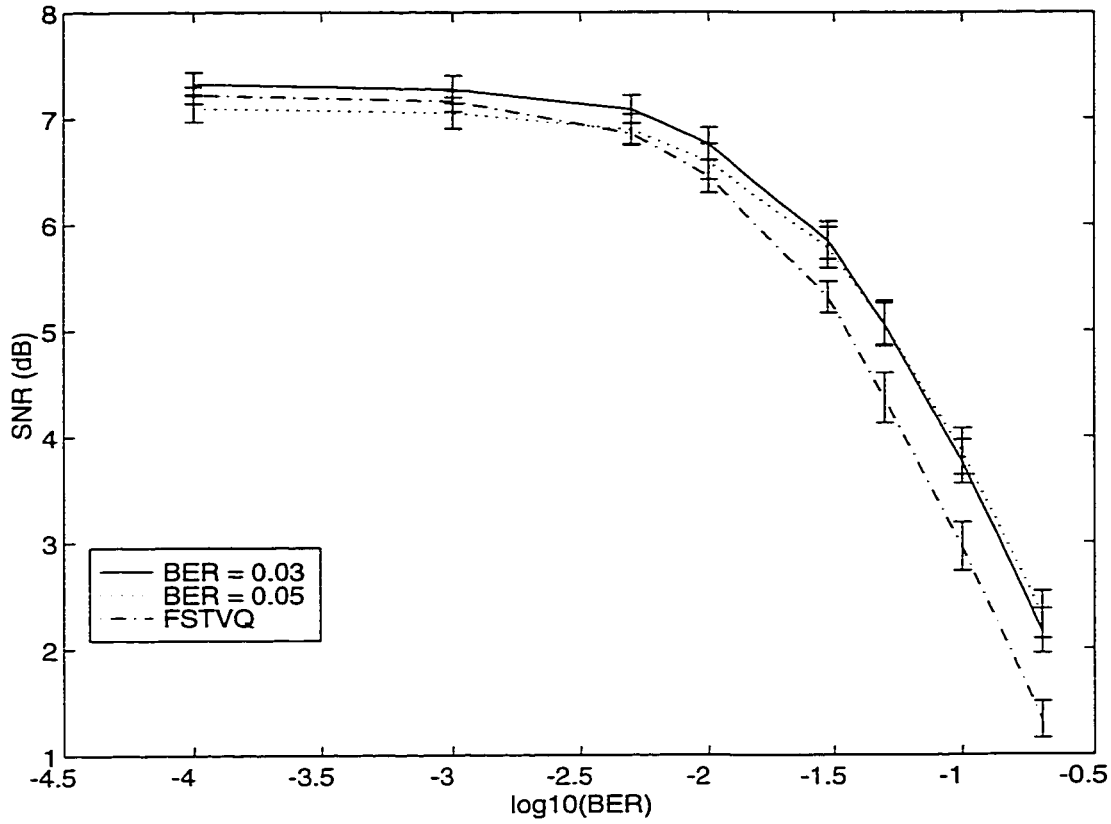

Figure 5.12: Channel mismatched performance using codebooks generated by CO-FSTVQ and FSTVQ. Two CO-FSTVQ codebooks are used; one associated with $\epsilon = 0.03$ and the other one with $\epsilon = 0.05$. The source is AR(2) and the compression rate is 0.5 bps with trellis parameters $k = 1$, $L = 2$ and $\nu = 5$.

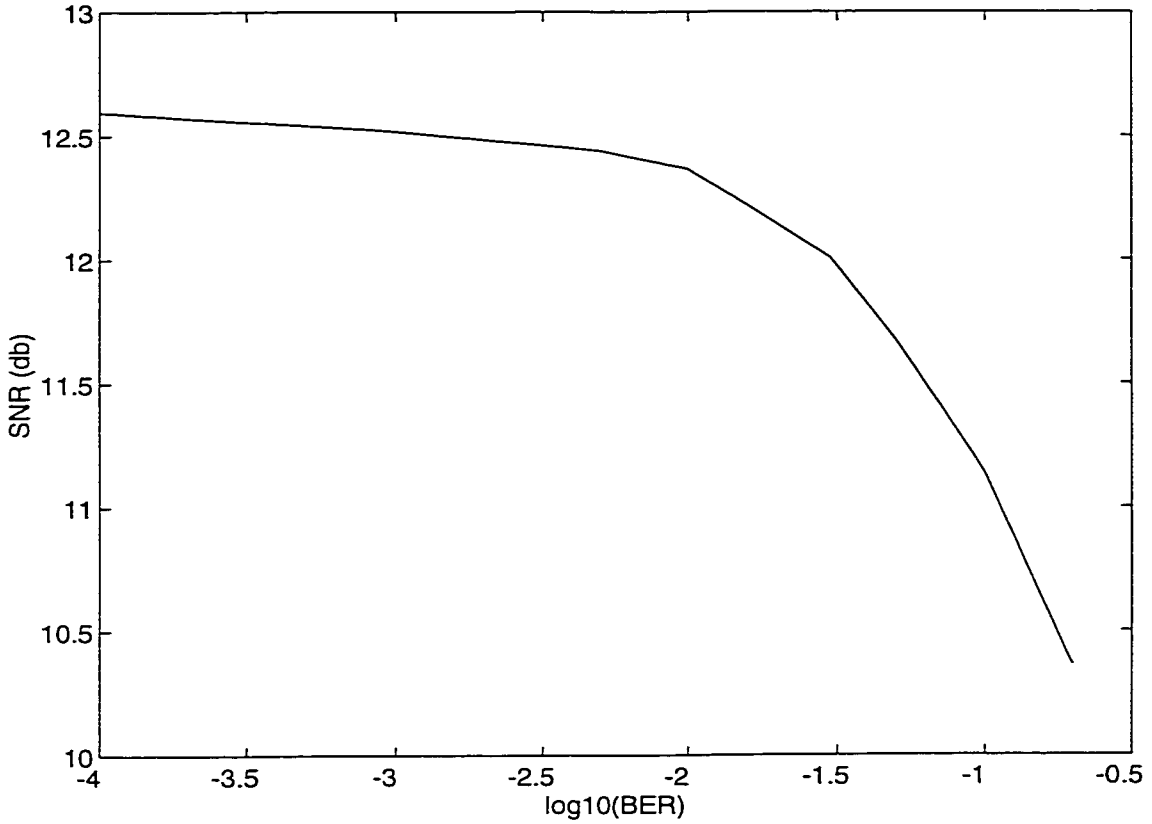

Figure 5.13: The MDTA curve for the AR(2) source.

$\epsilon_a \geq 0.05$. At the same time, the $\epsilon_d = 0.05$ codebooks under-perform the $\epsilon_d = 0.03$ codebooks for $\epsilon_a < 0.05$. However, the performance loss of adopting $\epsilon_d = 0.05$ for $\epsilon_a < 0.05$ is significantly larger than the gain over using $\epsilon_d = 0.03$ for $0.05 \leq \epsilon_a \leq 0.2$. Accordingly, a smart, yet not hard to make, decision should be taken regarding the choice of the proper ϵ_d according to the expected range of ϵ_a over the noisy channel.

Based on the summary presented in appendix A, we included the MDTA curves for AR(1) source in 5.10, 5.11. Note that the gap between the MDTA curves and the practically achieved curves is quite wide. Due to the large difference between the MDTA curve for the AR(2) source and the actual curves, and for the sake of clear presentation of the results, this MDTA curve is separately shown in Fig. 5.13. It is known that this large gap between the theoretical and the practical values can be bridged using higher compression rates, higher trellis constraint lengths as well as higher dimensions. Further improvement is expected by exploiting the source memory using for example predictive compression, as shown in Table XIX in [14] for the AR(2) source used in our case studies. However, it was observed in [22] that under channel-mismatched conditions predictive TVQ under-performs regular TVQ.

5.3.4 CO-TVQ versus Tandem Coding

At this point we are interested in showing where tandem source/channel coding systems come into the picture. Recall that the development of this kind of channel-optimized compression assumes a certain level of BER over the channel. Therefore, as far as the channel-optimized design is concerned, an error control coding system does only reduce the effective channel BER, and most of the times, it significantly reduces that value to very low levels. On the other hand, most of the channel-optimized quantization systems reported in literature provide "good" performance, that is worth the extra complexity, for $\epsilon_a \geq 0.01$. This can be demonstrated through the different simulation examples presented in this chapter. Thus, with a channel coding system we are actually better off with a SM-TVQ system rather than CO-TVQ. Nonetheless, as reported in [126], [24], using a tandem coding system involves extra delay, complexity as well as extra transmission bandwidth.

A fair comparison between a tandem coding system and a channel-optimized one should be based on the same transmission rate. For example, assume that we are to transmit the AR(1) source at a rate of 1 bps over a channel with $\epsilon = 0.0177$. One option is to first use a channel coding system, for instance a rate 1/2 convolutional code with a large constraint length, to reduce the channel BER to 0.0001. Then a SM-TVQ is used to perform a rate 1/2 compression, hence, delivering information

over the channel at 1 bps. As shown in Fig. 5.11, for $\nu = 2, k = 2, L = 4$, the expected SNR of this system is $\sim$ 8 dB. The exact same performance and transmission rate can be achieved just by using the codebook generated by CO-FSTVQ with $\nu = 2, k = 2, L = 2$ that is designed using $\epsilon_d = 0.03$, as shown in Fig. 5.10. Accordingly, we managed to achieve the same SNR performance and the same transmission rate using a much simpler joint system without the extra processing delay that is needed for channel decoding. Furthermore, based on the introduction argument of this chapter and as mentioned in [24], the expected performance of most tandem coding systems is quite optimistic compared to their actual practically-attainable performance.

5.4 Summary

This chapter introduced the FSTVQ approach in the context of channel optimized TVQ. We provided a review of the fundamentals of channel-optimized quantization along with some of the associated literature. This review showed the importance of channel-optimized compression as a joint source channel coding technique over tandem coding systems. The presentation started with a heuristic treatment of COVQ and CO-TVQ as an introduction to modifying the FSTVQ codebook search process to incorporate the effect of noisy channels. The modified equations provided a reliable trellis codebook search algorithm over noisy channels. Reliable in the sense of delivering lower distortion configurations using relatively short training sequences. Furthermore, the developed channel-optimized search process is significantly independent of the initialization step, which is usually a major problem associated with the channel-optimized LBG algorithm.

The new channel-optimized search algorithm (CO-FSTVQ) is implemented with the CO-LBG algorithm in several simulation cases using AR(1) and AR(2) Gauss-Markov sources. Based on these cases, we observed the following points:

- In general channel-optimized compression provides significant performance improvement over source-matched compression for channel BER $\epsilon \sim\geq 0.01$. Therefore, in cases with channel coding systems it is more efficient, in terms of complexity, to use source-matched encoders. However, note that extra complexity, delay and bandwidth are imposed on the overall system by using channel coding.

- The dependency of the CO-LBG algorithm on initial codebooks is a function of the level of channel noise. The dependency becomes less pronounced as the

channel noise increases. Nonetheless, compared to the performance of CO-FSTVQ, the performance of CO-LBG algorithm was significantly unstable for $\epsilon < 0.1$. The instability is measured by the difference between the minimum and maximum average SNR that are associated with the best and the worst minimum distortion configurations arrived at during 15 independent training trials.

- The robustness of CO-FSTVQ codebooks is tested under channel mismatched conditions compared to the performance of source-matched codebooks. The channel-optimized codebooks outperformed regular codebooks for values of $\epsilon_a \sim> 0.01$. Furthermore, simulation results showed that choosing the design BER ϵ_d is not a very critical issue especially at high channel BER. For example, the channel optimized codebooks associated with $\epsilon_d = 0.03$ and $\epsilon_d = 0.05$ had very close performance under channel mismatch conditions for $\epsilon_a \sim> 0.03$.

Chapter 6

Joint Source and Channel Decoding/Detection

An optimal source compression system is the one that produces redundancy-free finite-alphabet memoryless and uniformly-distributed sequences of symbols. The symbols at the output of ideal source encoders have the same significance or channel error sensitivity. However, residual redundancy is always expected at the channel input due to either source nonstationarity or suboptimal compression. The redundancy can be either correlation-related or distribution-related, as explained in appendix B. In general, redundancy removal, performed by data compression systems, forces the transmitted data to be more vulnerable to channel noise. In order to enhance the quality of transmission, channel coding is used to achieve the required reliability within minimum redundancy as well as reasonable complexity. Whether redundancy-management is performed in one combined source/channel system or two separate systems, the ultimate goal is to meet the channel capacity limitation within an expected average distortion of reproduction. Although using two separate systems of compression and channel coding seems rather unnecessary, in its natural form, the redundancy in the information sequence is kind of raw or unrefined. One method for a combined system was introduced in chapter 5, where compression is performed considering the channel distribution, which made the source encoder output sequence less vulnerable to channel noise.

Another form of joint source and channel (JSC) coding is performed using the source *a priori* information, at the channel decoder input, in a JSC decoding/detection approach [34], [33], [18] and [35]. The JSC decoder uses the *a priori* information, which is inherent in the redundancy statistics, to enhance the reliability of decisions. The approach is simply a MAP decoding/detection procedure. The authors in [34], [18] and [35] use the residual redundancy at the channel input for JSC sequence estimation. In [34], the residual redundancy at a DPCM source encoder output is used, while in [18] and [35] the authors consider the detection of Markov channel input sequences. On the other hand, the authors in [33] develop a JSC sequence-MAP convolutional channel decoder. The study in [33] addresses the problem of JSC decoding of binary as well as nonbinary convolutional codes using the residual redundancy at the output of DPCM-compressed images.

The decoding schemes addressed in [34] and [33] are based on hard sequence-MAP decoding. In [18] and [35], detection is based on hard sequence-MAP detection as well as a new instantaneous symbol- MAP algorithm. In this chapter we extend the work established for JSC MAP decoding through several case studies [128], [129] and [130]. We shall develop the soft decoding versions of the existing MAP approaches. Furthermore, we will test the effect of having different redundancy (correlation) levels, at the source encoder output, on the decoding gain. An improved performance is achieved in [33] using nonbinary convolutional codes, which match the source output alphabets. In this chapter, we will also demonstrate the advantage of using the simple dual-k nonbinary convolutional codes in providing improved levels of JSC error protection compared to binary convolutional codes. For the sake of completeness, appendix E is introduced to provide a comparative study between symbol-MAP, sequence-MAP as well as instantaneous-MAP detection. The appendix shows the superiority of symbol-MAP detection in providing lower distortions of reproduction at comparable bit-error rates. This expected behaviour is due to the fact that maximizing the probability of correctly decoded symbols reflects into maximizing the probability of correct reproduction codewords.

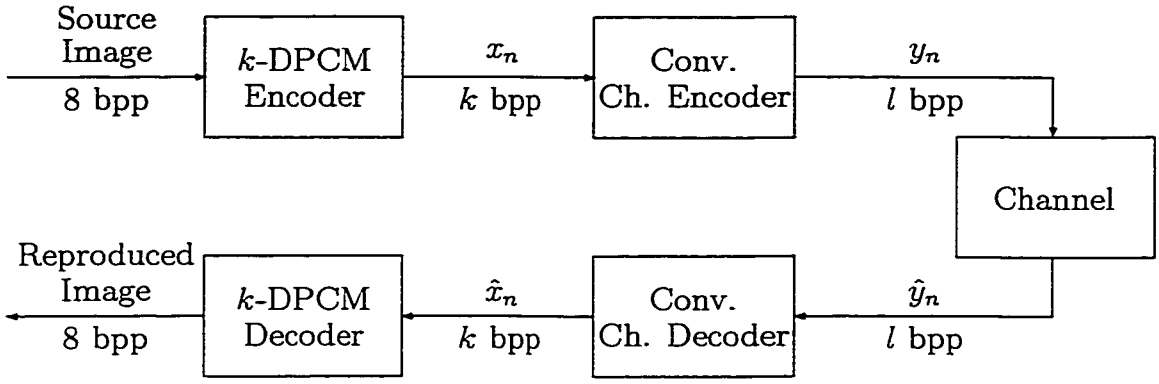

Figure 6.1: The communications system under consideration.

6.1 JSC Soft Viterbi Decoding

Based on maximizing the probability of correct decision, the authors in [33] and [34] use the residual correlation at a DPCM source encoder output to derive a new path metric for a joint source/channel (JSC) hard-decision sequence-MAP Viterbi decoder. The developed decoder is shown to outperform the conventional hard-decision ML-Viterbi decoder at relatively low SNR values. In this section we extend the derivation presented in [33] to the case of soft-decision sequence-MAP Viterbi decoding. The advantage of soft JSC-decoding is demonstrated via simulating rate 2/3, 1/2, 3/4, dual-2 as well as dual-3 convolutional codes with DPCM-compressed gray images. We will show the significance of matching the alphabets of the source encoder output to the channel encoder input in providing better error protection. The advantage of higher levels of residual redundancy in JSC-decoding is demonstrated using 3-bit DPCM. This source encoder provides higher levels of nonuniformity, which is measured by the difference between the number of bits per symbol and the symbol entropy.

6.1.1 System Configuration

The system under consideration includes a source of information, a DPCM system, a memoryless Gaussian channel and a channel coding system as shown in Fig. 6.1. The source of information is a still gray image, which is represented by 8 bits/pixel (bpp). The k-DPCM system includes both a 5th-order linear predictor (LP) matched to the source image, and a pdf-optimized uniform quantizer with 2^k reproduction levels. The reader is referred to [50] for a complete treatment of the structure as well as the operation of DPCM systems. The pdf-optimized quantization is

Table 6.1: The DPCM system parameters and statistical measures.

	2-DPCM	3-DPCM
Δ_{opt}/σ_e	1.0874	0.7309
μ_e	−0.00671	−0.00457
σ_e^2	0.115893	0.093252
Q Lower Limit	−0.372	−0.672
Δ	0.372	0.224
PSNR (dB)	27.538	31.134
$H(x)$	1.528 bit/sym	2.009 bit/sym

performed using a midrise quantizer along with the normalized step size (Δ_{opt}/σ_e) values tabulated in Table 4.1 of [41]. The symbol σ_e stands for the standard deviation of the sequence that represents the difference between the linear predictor output and the image pixels. It is known that this error sequence is best modeled by a Laplacian distribution [131]. In Table 6.1 we present the different parameters necessary to perform 2-DPCM and 3-DPCM on the training image as well as the over all PSNR values. In this table **Q** is used to denote the midrise quantizer.

Compression is performed by delivering k-bit output symbols $\{x_n\}_{n=0,1,...}$ for the corresponding 8-bit pixels. As shown in Table 6.1, the symbol entropy at the DPCM encoder output is found to be equal to 1.528 bits/symbol and 2.009 bits/symbol for 2-DPCM and 3-DPCM, respectively. This means that the amount of redundancy due to nonuniformity provided by the 3-DPCM system is 0.991 bits/symbol, while the 2-DPCM system provides 0.472 bits/symbol. Furthermore, higher degrees of correlation (memory) is expected at the output of the 3-DPCM encoder, due to the increased number of reproduction symbols as compared to 2-DPCM. Therefore, the residual redundancy using 3-DPCM is higher than that left at the output of the 2-DPCM encoder. We shall demonstrate in the simulations section how to use this redundancy to provide higher levels of error protection.

On the other hand, the convolutional encoder configuration employed in our simulations is the one presented in [124]. The encoder has a rate $R = k/l$ and a constraint

length ν, thus a decoding trellis of $2^{k(\nu-1)}$ possible states with 2^k branches out of each node (state). In the simulations section we shall use the following convolutional codes

- $R = 1/2$, $\nu = 3$, $d_{free} = 4$, with $G = [4, 7]$.
- $R = 2/3$, $\nu = 3$, $d_{free} = 5$, with $G = [27, 75, 72]$.
- $R = 3/4$, $\nu = 2$, $d_{free} = 4$, with $G = [13, 25, 61, 47]$.
- Dual-2, $\nu = 2$, $d_{free} = 4$, with $G = [12, 5, 16, 11]$.
- Dual-3, $\nu = 2$, $d_{free} = 4$, with $G = [44, 22, 11, 64, 12, 41]$.

where G is the generator polynomial in octal form. The transmission is BPSK over an AWGN channel. Therefore, the energy per bit required to maintain a specific bit error rate ϵ is

$$E_b = \frac{N_o}{R} \left[\mathcal{F}^c(1 - 2 \times \epsilon)\right]^2 \tag{6.1}$$

where $\mathcal{F}^c(\cdot)$ is the complementary error function, and $N_o/2$ is the AWGN two-sided spectral density.

6.1.2 Soft Sequence-MAP Viterbi Decoding

In harmony with the thesis notation, let the source encoder output be the stationary discrete-time sequence $\{x_n\}_{n=1}^{\infty}$ with alphabet $\mathcal{A} \equiv \{0, 1, 2, \ldots, I-1\}$. Furthermore, we will denote the channel input sequence by $Y = \{y_0, y_1, ..., y_{L-1}\}$ and its output sequence by $\hat{Y} = \{\hat{y}_0, \hat{y}_1, ..., \hat{y}_{L-1}\}$, where L represents the sequence length, and both $\hat{y}_n$ and y_n are l-bit symbols. The authors in [33] use the residual redundancy (correlation) at the DPCM encoder output, or equivalently at the channel input, to propose their modified hard decision JSC (sequence-MAP) Viterbi metric. This correlation is modeled by a first order Markov process. The optimum MAP decoder maximizes the probability of correct decision according to [33]

$$\max\left\{\log\{P(\hat{Y}|Y)P(Y)\}\right\} = \max\left\{\sum_{n=n_0}^{n_0+L-1} \log\{P(\hat{y}_n|y_n)P(y_n|y_{n-1})\}\right\} \tag{6.2}$$

Using the Viterbi algorithm to implement this MAP-decoding rule involves deciding on the branch index $x_{n=n_0}$, which is associated with the best metric at state s_{n_0+L-1}. Accordingly, the Viterbi algorithm follows the decoder trellis using the branch metric

$$M_n^m = \max_{x_n, s_{n-1}} \left\{ M_{n-1}^{\acute{m}} + \log P(y_n|y_{n-1}) + \sum_{i=0}^{l-1} [\log P(\hat{y}_{n,i}|y_{n,i})] \right\} \tag{6.3}$$

where y_n is the channel input symbol associated with branch index x_n and $s_n = m$, and $m = 0, 1, \ldots, 2^{k(\nu-1)} - 1$. Besides, y_{n-1} is the channel input symbol that colours the branch connecting state $s_{n-1} = \acute{m}$ and state $s_n = m$. The hard decision decoding version of the metric in Eq. (6.3) is given by [33]

$$M_n^m = \max_{x_n, s_{n-1}} \left\{ M_{n-1}^{\acute{m}} + \log P(y_n|y_{n-1}) + \log P(\hat{y}_n|y_n) \right\} \tag{6.4}$$

For the binary symmetric channel, the value of $P(\hat{y}_n|y_n)$ is evaluated according to

$$P(\hat{y}_n|y_n) = (\epsilon)^{h_d}(1-\epsilon)^{l-h_d}$$

where h_d is the Hamming distance between $\hat{y}_n$ and y_n.

In order to derive the soft decoding metric, we use the channel Gaussian distribution

$$P(\hat{y}_{n,i}|y_{n,i}) = \frac{1}{\sqrt{\pi N_o}} \exp\left\{ -\frac{(\hat{y}_{n,i} - \sqrt{RE_b}y_{n,i})^2}{N_o} \right\}; \quad i = 0, 1, \ldots, l-1$$

where $R = k/l$ is the channel coding rate, RE_b is the energy per transmitted bit, and E_b is given by Eq. (6.1). Substituting for the channel distribution in Eq. (6.4) produces

$$M_n^m = \max_{x_n, s_{n-1}} \left\{ M_{n-1}^{\acute{m}} + \frac{N_o}{RE_b} \log P(y_n|y_{n-1}) - \sum_{i=0}^{l-1} \left[\left(\frac{\hat{y}_{n,i}}{\sqrt{RE_b}} - y_{n,i} \right)^2 \right] \right\} \tag{6.5}$$

or equivalently

$$M_n^m = \max_{x_n, s_{n-1}} \left\{ M_{n-1}^{\acute{m}} + \frac{N_o}{RE_b} \log P(y_n|y_{n-1}) + \sum_{i=0}^{l-1} \left[\frac{2 \cdot \hat{y}_{n,i} \cdot y_{n,i}}{\sqrt{RE_b}} \right] \right\} \tag{6.6}$$

Eq. (6.6) represents the soft version of the JSC metric developed in [33].

6.2 Performance Evaluation

The implementation of the soft decision JSC metric (Eq. (6.6)) and the hard decision metric (Eq. (6.4)) requires first estimating the term $P(y_n|y_{n-1})$ at the channel input. Note that this probability measure is both source and channel encoder dependent. However, for ease of implementation, if we assume stationarity in the source statistics, we may use the same estimate for similar images [33]. For a more involved approach of an online parameter estimation, the authors in [132] use a forward/backward algorithm to estimate the Markov model probability variables, which describe the redundancy at the channel input. The simulations in this section consider computing the bit error rate, which represents the ratio of decoded information bits in error to the total number of information bits. Following a Poisson approximation, the 95% confidence interval is presented as well for the different results introduced in this section. We shall consider the following two case studies.

6.2.1 Hard versus Soft JSC Decoding

In Figs. 6.2 and 6.3 we demonstrate the performance of both the rate 2/3 code and the dual-2 code with conventional soft decision decoding (SDD) and JSC-SDD. Note that in Fig. 6.3 the JSC decoding metric improves the performance of the dual-2 nonbinary code for the entire range of the relatively low SNR. On the other hand, we can see in Fig. 6.2 the point after which SDD outperforms JSC-SDD decoding. This behaviour is also observed in [33] and [1]. This simply means that even though the channel input exhibits statistical correlation, we are better off using ML decoding during "good" channel conditions. In order to clarify this behaviour, note that the JSC decoding metric given by Eq. (6.6) consists of two terms: the weighted channel input *a priori* information and the channel-dependent *a posteriori* information. For good channel conditions, note that the spectral height of the channel noise N_o is "small", which accentuate the contribution of the channel-dependent part of Eq. (6.6). Under these conditions, it is enough to use the error-correcting capability of the channel coder along with the received channel output symbols to provide a reliable decision. In fact, using the extra *a priori* information might produce misleading results after a critical E_b/N_o value, as shown in Fig. 6.2. The critical E_b/N_o value is clearly dependent on the characteristics of the channel coding system. The author in [1] suggests using the source *a priori* information (MAP-decoding) only during bad channel conditions (low SNR, or high N_o).
On the other hand, the expected improvement of using JSC-SDD over JSC hard decision decoding (HDD) is demonstrated in Figs. 6.4 and 6.5.

6.2.2 Source-Matched JSC Coding

Although the sequence of bits at the source encoder output exhibits certain level of residual redundancy, the "actual" information, and thus redundancy, is conveyed through the symbols at the encoder output. Therefore, the residual redundancy is best exploited by symbol-based processing rather than a bit-wise treatment. This intuitively suggests matching the source encoder output alphabets to the channel encoder input, which reflects into better mapping of the residual redundancy of the source encoder output to the channel input. From a channel decoding point of view, this matching procedure has a significant impact on the performance of the JSC decoding process. As shown in Eq. (6.6), the JSC decoding metric includes the correlation measure $P(y_n|y_{n-1})$, which is directly related to the residual redundancy at the channel input. Source-matched channel coding corresponds to using the source encoder output symbols as indices for the decoding trellis branches. Note that the correlation between successive branch labels (y_n) is directly related to the correlation between consecutive branch indices. Accordingly, using source-matched channel coding results in exploiting the available residual redundancy levels more efficiently through the JSC decoding metric.

A new family of nonbinary convolutional codes is proposed in [33] for the purpose of source-matched JSC decoding. The performance of these codes is close to binary convolutional codes, which are matched to the source output alphabets. In this section we demonstrate the impact of using the exact redundancy structure available at the source encoder output in providing better JSC error protection. For this purpose we present the following two cases using 2-DPCM and 3-DPCM systems. Recall that due to increasing the compression rate, higher levels of redundancy are available at the 3-DPCM encoder output. Accordingly, improved JSC decoding performance is expected using 3-DPCM.

Case 1: Rate 1/2 Channel Code with 2-DPCM vs. Rate 3/4 Channel Code with 3-DPCM

Fig. 6.6 represents the performance results of using rate 1/2 channel code with 2-DPCM, and rate 3/4 channel code with 3-DPCM. Note that the two systems have the same transmission rate of 4 bits/sample. The rate 1/2 code outperforms the rate 3/4 code at low SNR with SDD, while at "high" SNR the rate 3/4 code provides better performance. Nonetheless, the rate 3/4 with 3-DPCM system provided better error protection with JSC decoding. The improvement in performance is attributed to the fact that the input of the rate 3/4 encoder is matched to the DPCM encoder output. Meanwhile, the rate 1/2 channel code is using the redundancy in the sequence of bits rather than symbols at the 2-DPCM output, which does not properly

represent the residual compression redundancy at the source encoder output.

Case 2: 3-DPCM with Dual-3 vs. 2-DPCM with Dual-2 Code.

In this case we demonstrate the effect of exploiting higher redundancy levels to provide better JSC error protection. Furthermore, we show the significance of source-matched channel coding on the proper performance of the JSC decoding process. The benefit of source-matched channel coding is depicted in Fig. 6.7, where the rate 1/2 dual-k channel codes with k-DPCM outperform the rate 1/2 channel code with 3-DPCM. As shown in Fig. 6.8, dual-3 and dual-2 codes provide comparable bit error rate performance with soft decision decoding. However, due to the increased amount of residual redundancy at the 3-DPCM encoder output, the dual-3 code provides significant coding gain with JSC-SDD compared to the performance of dual-2/2-DPCM system, as shown in Fig. 6.7. Nevertheless, we should keep in mind that the increased redundancy is associated with a 3/2 factor of increased transmission rate.

In order to see the impact of the channel coding part on the perceptual quality of the reproduced image, we chose the channel with $E_b/N_o = 4.86$ dB. This point represents the best channel conditions under consideration that is associated with the rate 1/2 channel codes. To start with, we show in Figs. 6.9 and 6.10 the error-free reconstructed image using 2-DPCM and 3-DPCM, respectively. As shown in these two figures, using 3-DPCM provides a gain of 3.596 dB of PSNR. Decoding the compressed images at the output of the noisy channel without error protection produces the images shown in Figs. 6.11 and 6.12. It is clear that the image quality is highly deteriorated at this channel BER of 0.04, which is computed using Eq. (5.18). In Fig. 6.13 we show the effect of using dual-2 channel code with SDD, which provided a BER of 2.29×10^{-4} accompanied with a PSNR of 25.712 dB. An improvement in the image quality associated with a PSNR of 27.023 is achieved using the Dual-2 code, however, with JSC-SDD. The MAP decoding metric provided and average BER of 7.63×10^{-5}. Similarly, Figs. 6.15 and 6.16 show the advantage of using JSC-SDD over SDD with the dual-3 channel code and 3-DPCM. The subjective improvement is associated with a PSNR gain of 1.847 dB. Furthermore, this improvement resulted from an average BER of 1.91×10^{-5} as opposed to 1.14×10^{-4} for dual-3 JSC-SDD and SDD, respectively. The different performance measures for this case are tabulated in Table 6.2.

Table 6.2: The overall system performance results over the channel with $E_b/N_o = 4.86$ dB. The channel codes used are dual-2 and dual-3 codes with 2-DPCM and 3-DPCM, respectively.

	2-DPCM		3-DPCM	
	PSNR (dB)	BER	PSNR (dB)	BER
Error Free Channel	27.538	0.0	31.134	0.0
W/O Channel Coding	9.264	0.04	8.458	0.04
JSC-SDD	27.023	7.63×10^{-5}	29.543	1.91×10^{-5}
SDD	25.712	2.29×10^{-4}	27.696	1.14×10^{-4}

6.3 Summary

This chapter presented an extension to the work developed for joint source and channel decoding, [33]. The studied system involved transmitting a DPCM encoded image over a memoryless Gaussian channel with convolutional channel codes. After introducing the system under consideration, we presented the derivation for a soft JSC MAP decoding metric, in accordance with the hard metric introduced in [33]. The JSC decoding metric was developed to take advantage of the residual redundancy left at the source encoder output. The advantage of using the soft decision metric over the hard decision metric was shown through the different studied examples. Simulation cases pointed out the following observations

- The advantage of using JSC decoding over conventional ML decoding at low SNR values, in particular we showed the expected advantage of soft over hard channel decoding.
- The achieved gain of using source-matched channel codes in providing lower JSC decoding bit error rates. This point was tested using dual-k convolutional channel codes matched to the k-DPCM compression systems.
- We managed to achieve improved coding gains using increased levels of residual redundancy. We showed that in order to properly exploit the advantage of having higher levels of residual redundancy, the source encoder output alphabets should be matched to the channel encoder input.

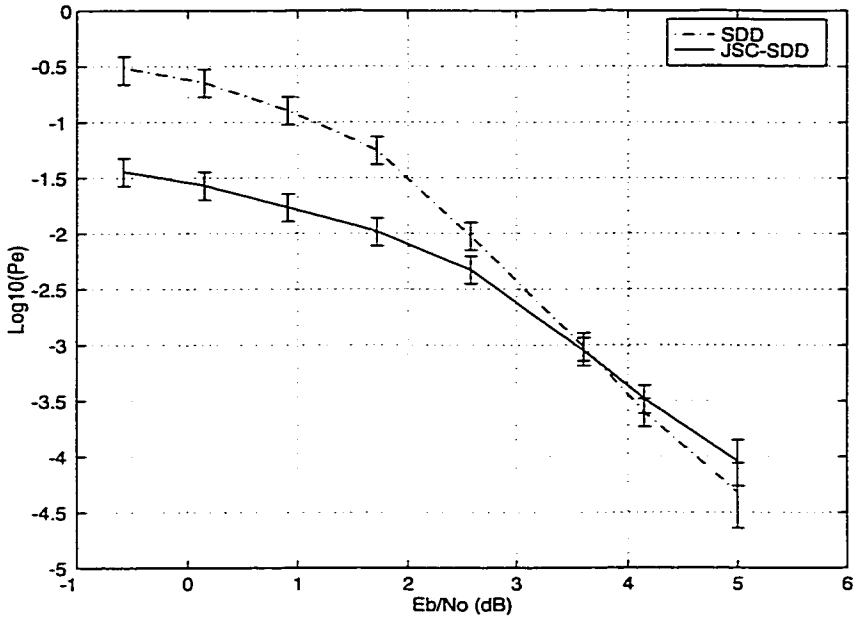

Figure 6.2: The performance of the rate 2/3 code.

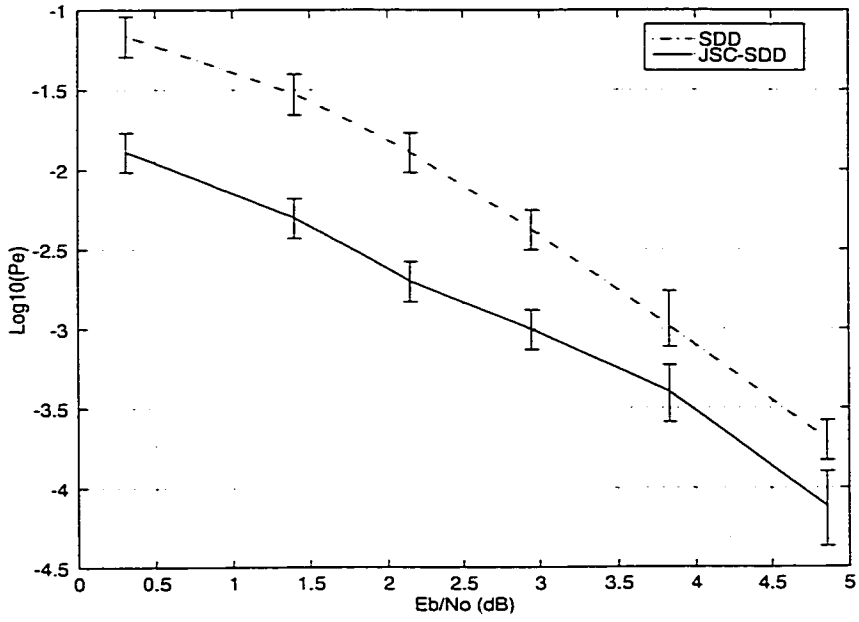

Figure 6.3: The performance of the dual-2 code.

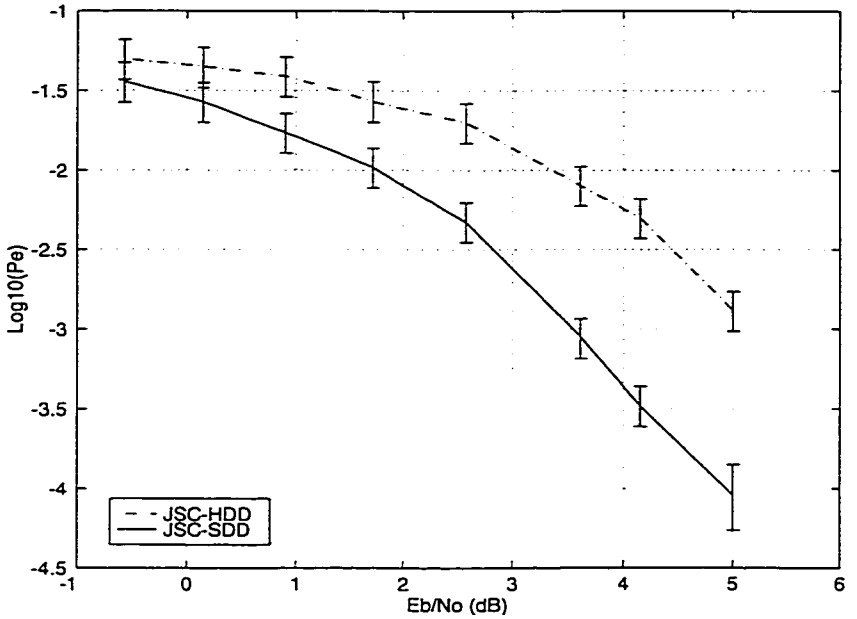

Figure 6.4: $R = 2/3$ code: hard vs. soft JSC decoding.

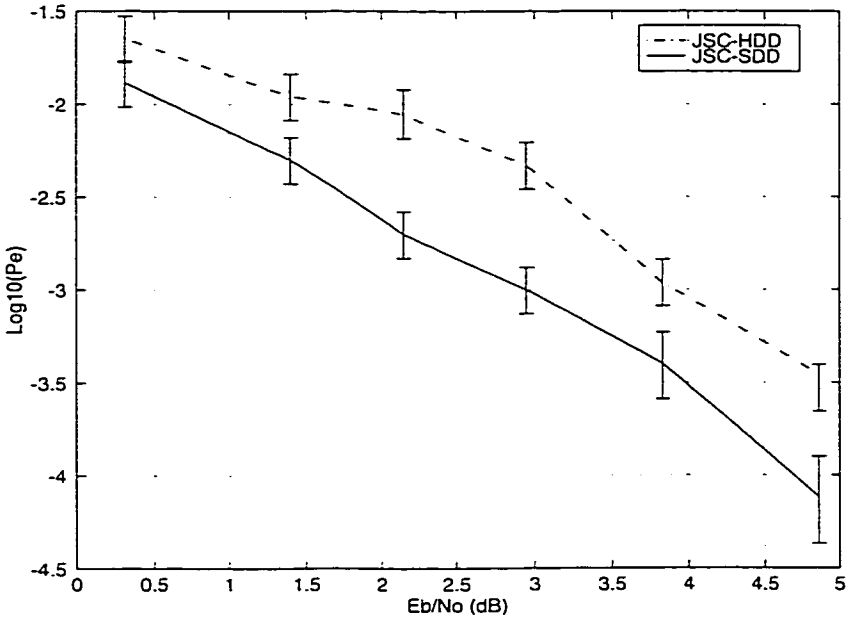

Figure 6.5: Dual-2 code: hard vs. soft JSC decoding.

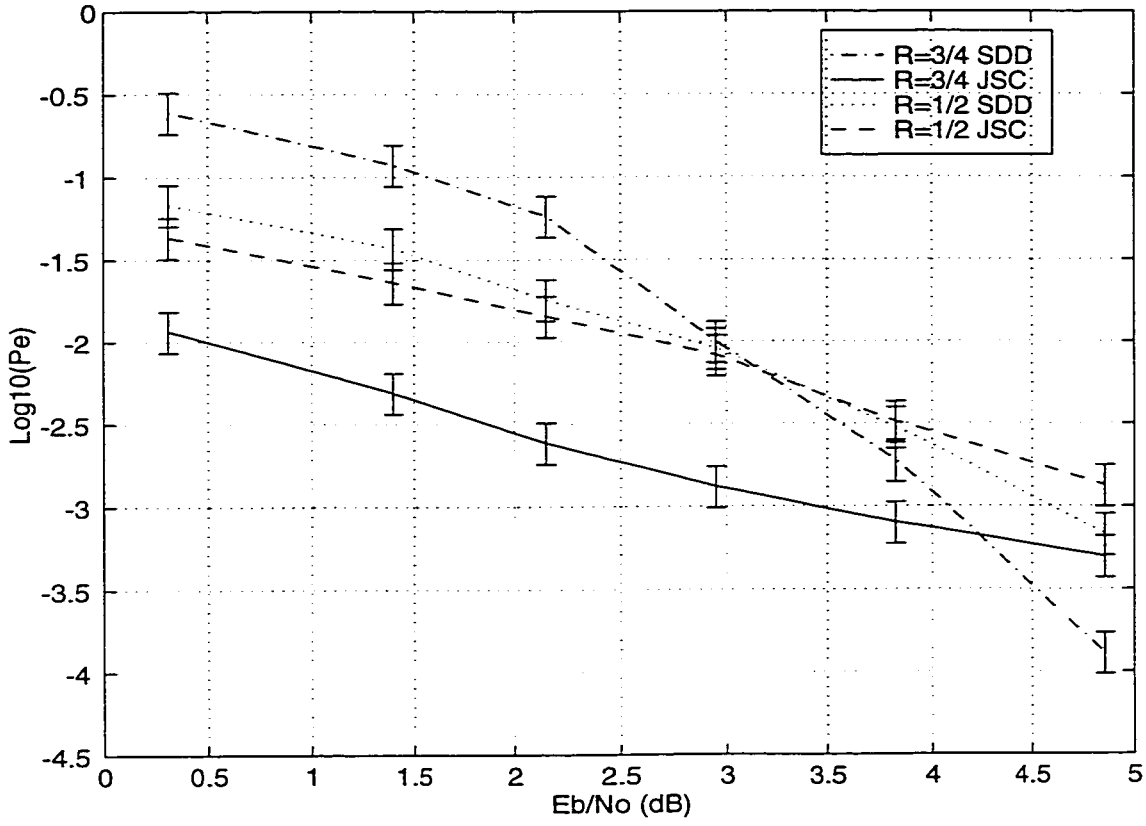

Figure 6.6: 3-DPCM with rate 3/4 code vs. 2-DPCM with rate 1/2 code.

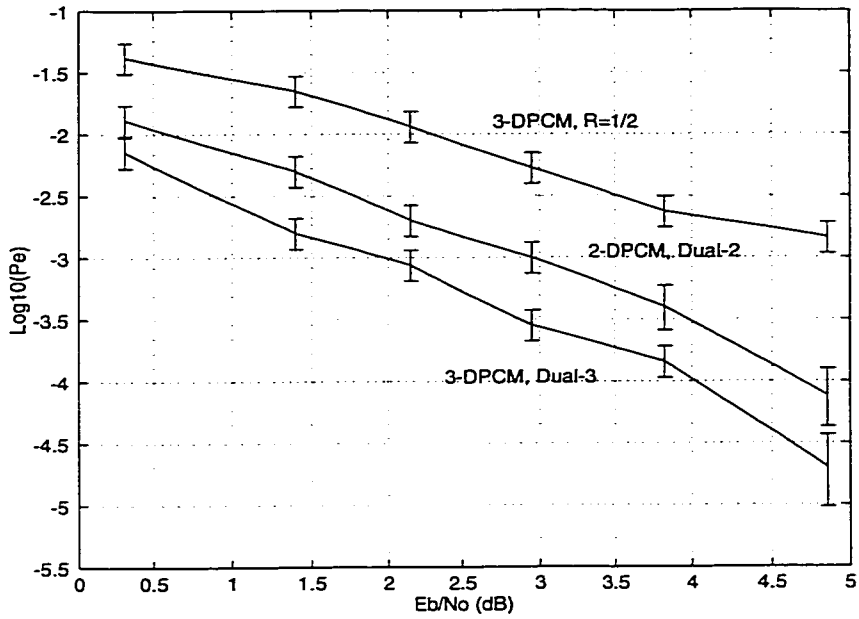

Figure 6.7: JSC soft decoding: 3-DPCM with rate 1/2 and dual-3 code and 2-DPCM with dual-2 code.

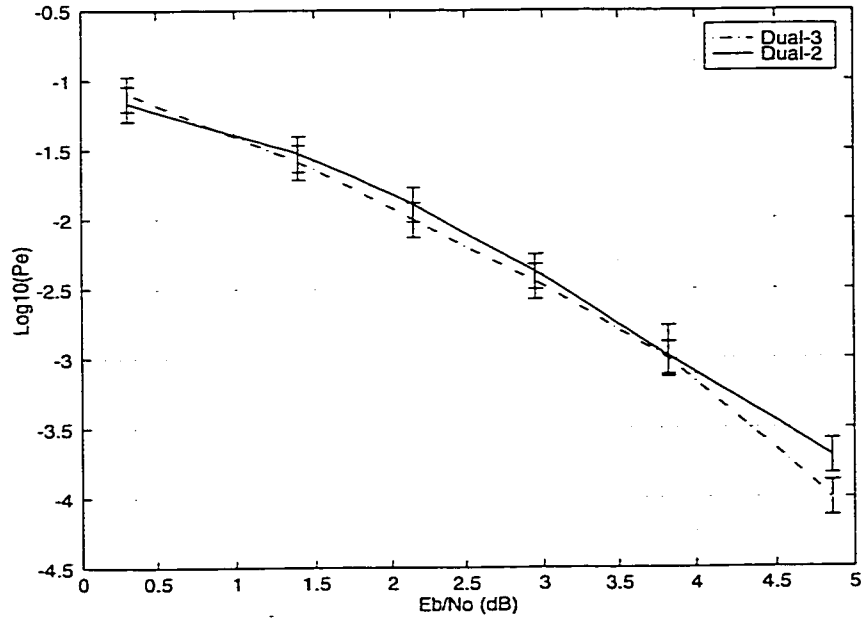

Figure 6.8: Soft decision decoding: dual-2 with 2-DPCM vs. dual-3 with 3-DPCM.

Figure 6.9: The 2-DPCM reconstructed image over an error-free channel. PSNR = 27.538 dB.

Figure 6.10: The 3-DPCM reconstructed image over an error-free channel. PSNR = 31.134 dB.

Figure 6.11: The 2-DPCM reconstructed image over a channel without error-protection. The channel has an $E_b/N_o = 4.86$ dB, thus BER $= 0.04$. PSNR $= 9.2639$ dB.

Figure 6.12: The 3-DPCM reconstructed image over a channel without error-protection. The channel has an $E_b/N_o = 4.86$ dB, thus BER $= 0.04$. PSNR $= 8.4576$ dB.

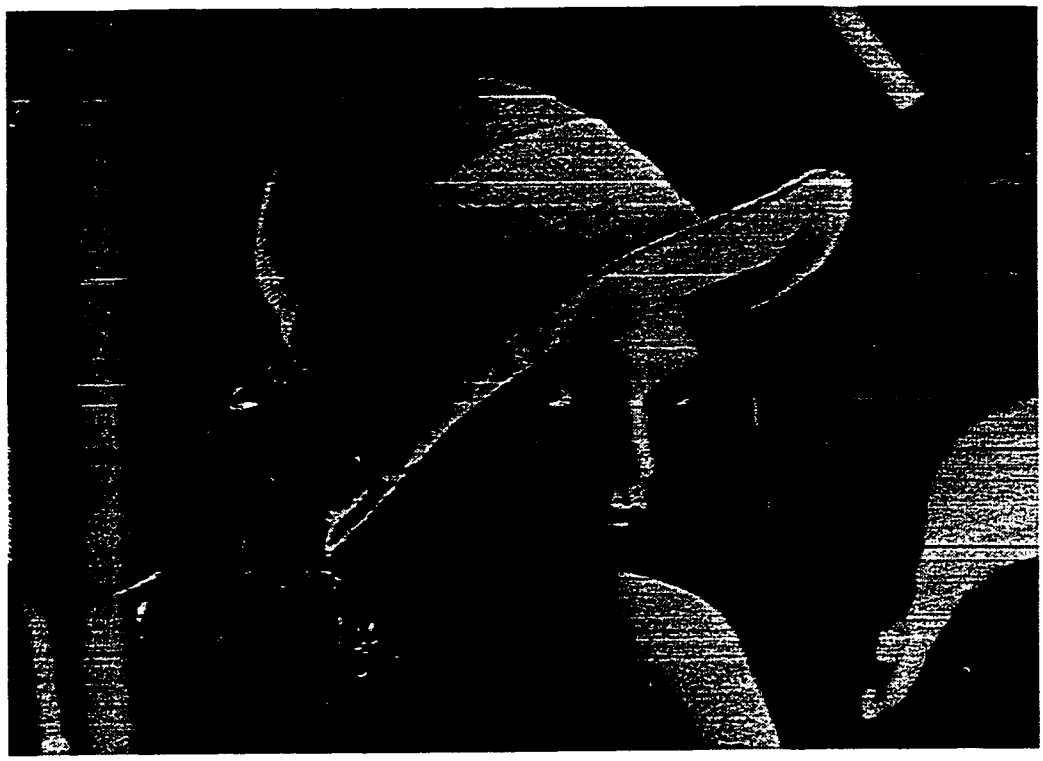

Figure 6.13: The 2-DPCM reconstructed image over a channel with Dual-2 code and SDD. The channel has an $E_b/N_o = 4.86$ dB, the channel code provides an average BER of 2.29×10^{-4}. PSNR = 25.712 dB.

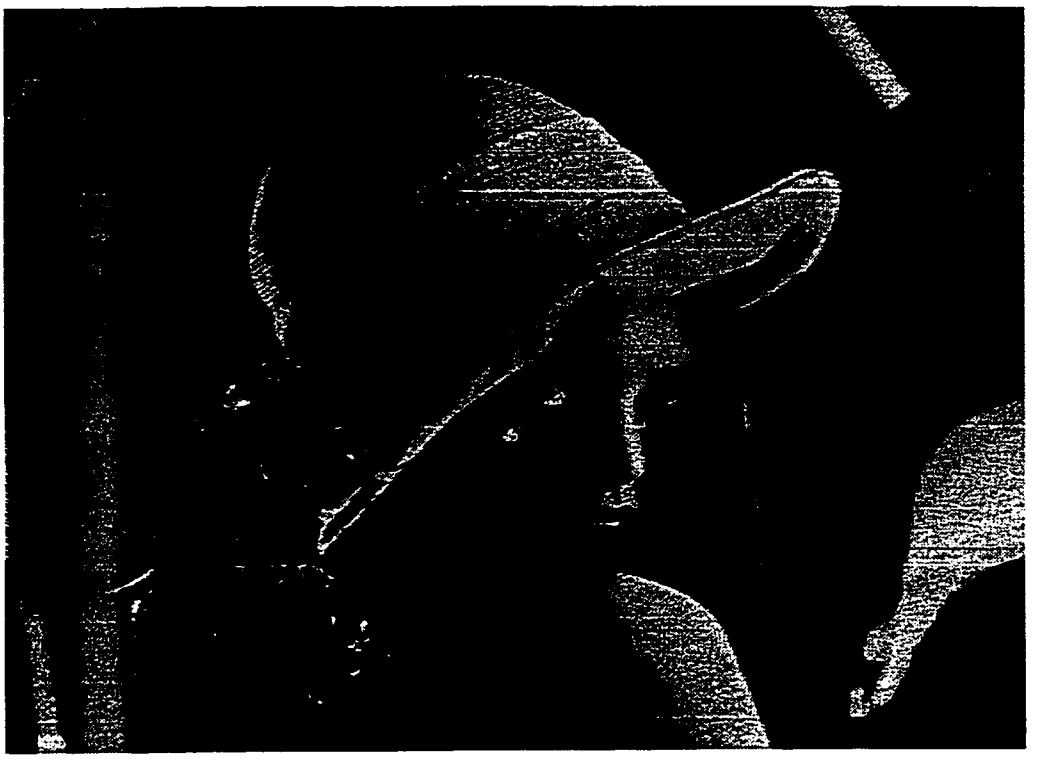

Figure 6.14: The 2-DPCM reconstructed image over a channel with Dual-2 code and JSC-SDD. The channel has an $E_b/N_o = 4.86$ dB, the channel code provides an average BER of 7.63×10^{-5}. PSNR = 27.023 dB.

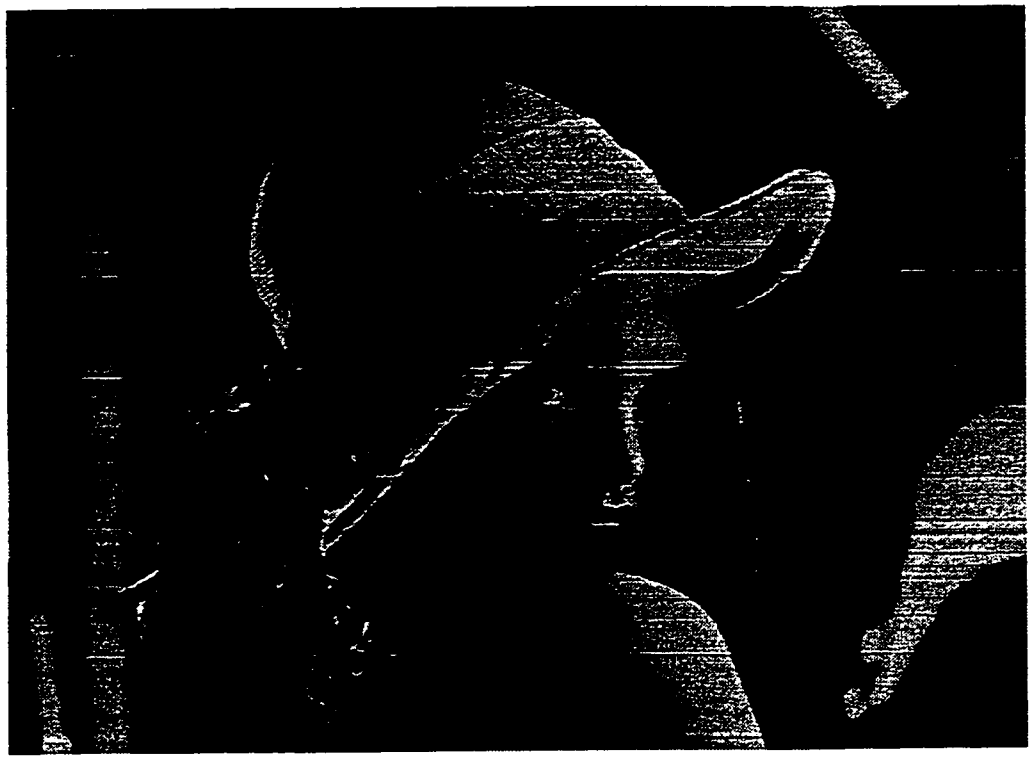

Figure 6.15: The 3-DPCM reconstructed image over a channel with Dual-3 code and SDD. The channel has an $E_b/N_o = 4.86$ dB, the channel code provides an average BER of 1.14×10^{-4}. PSNR = 27.696 dB.

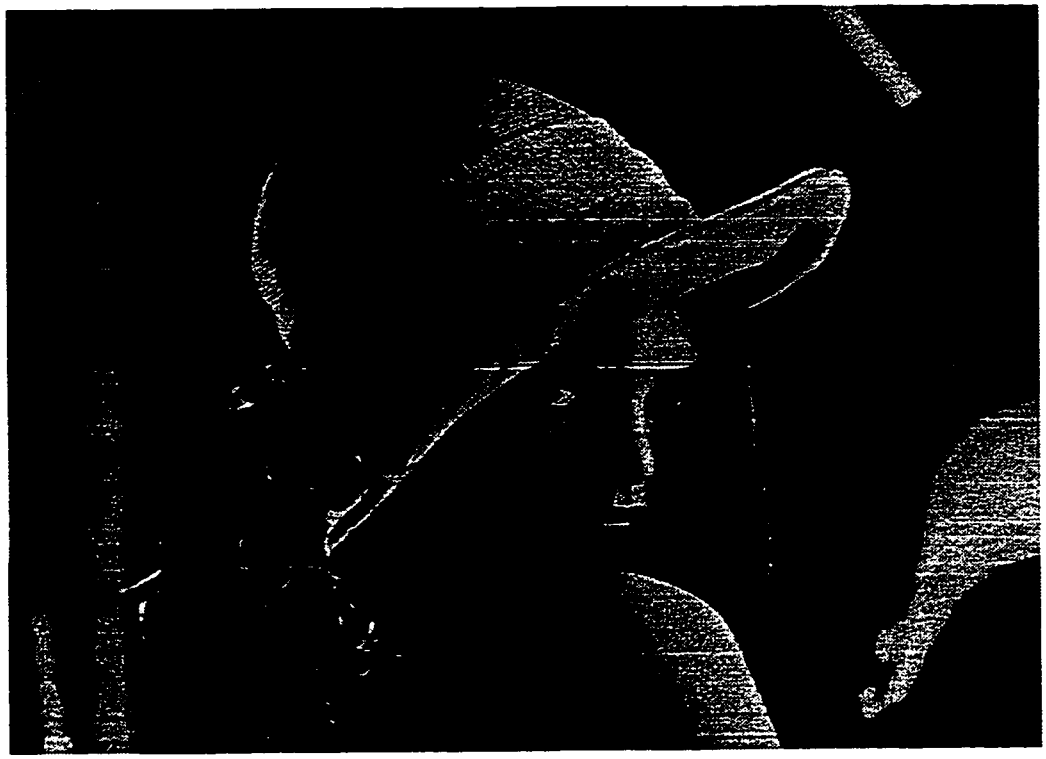

Figure 6.16: The 3-DPCM reconstructed image over a channel with Dual-3 code and JSC-SDD. The channel has an $E_b/N_o = 4.86$ dB, the channel code provides an average BER of 1.91×10^{-5}. PSNR = 29.543 dB.

Chapter 7

Conclusions

This chapter rounds up the major research highlights of this thesis. The general concepts treated in this thesis are related to trellis data compression and joint source/channel coding. Our presentation started with introducing the preliminary background necessary for understanding VQ-related topics as well as soft-output channel decoding. Chapter 2 shed some light into the basics of VQ and TVQ including design and implementation issues. Furthermore, we presented deterministic annealing as a VQ codebook search technique. In Chapter 3 we introduced the basic idea of soft output channel decoding as well as related algorithms. The introduction included a heuristic treatment of the symbol-MAP soft-output channel decoding algorithm. These two chapters established the proper framework, within which we developed the material presented in the subsequent chapters.

The first major contribution in this thesis was using the concept of soft output channel decoding in the area of data compression. The development was established considering the source of information as a communication channel, however, with a distortion-based statistical distribution model. This treatment of the information source paved the way to applying the forward-backward symbol-MAP algorithm for data compression. This algorithm was employed with TVQ as a search algorithm for the minimum distortion path, and it was denoted by the soft TVQ algorithm (STVQ). By soft we refer to the feature of delivering a set of reliability information, which describes the likelihood of all possible TVQ output symbols in the minimum distortion sense. The overall performance of the minimum distortion symbols delivered by the STVQ algorithm was shown to be within the proximity of the performance achieved by the Viterbi algorithm with TVQ. Furthermore, we showed that

the performance of the STVQ algorithm is closely related to a Lagrangian multiplier (η), which controls the degree of association of a source output vector to a quantization cell represented by a reproduction codevector. In reference to the problem of channel decoding, the parameter η is very similar to the SNR variable, which appears in the channel distribution function. The value of SNR is usually known and controlled by the power of transmission in addition to the level of channel noise. This fact reflects into a clear description of the decoding space, which leads to the accurate computation of the likelihood of a channel output symbol as compared to the different channel hypotheses values. On the other hand, the value of η was experimentally shown to be related to the anticipated value of average distortion of reproduction $D(R)$. Note that this value is clearly related to the source variance and trellis structure.

Introducing STVQ represents a building block, which sets the ground work for several future potential source compression applications. The attractive feature of delivering soft distortion-related reliability information might come in handy for multistage compression systems as well as joint compression and channel coding/modulation systems. In this context we mention residual quantization, in particular multistage trellis coded quantization systems, [133] and [134]. Passing soft information along with hard decisions from one stage to another might provide improved mapping of the information sequence using the trellis as well as the residual codebooks structures. Furthermore, the STVQ algorithm might provide significant advantage when applied for joint optimization of trellis quantization/modulation [135]. To get the sought for advantage of applying the STVQ algorithm in multistage source coding systems, the STVQ algorithm has to be modified to make use of soft information in subsequent stages. Some insight and lessons regarding such applications can be learned from iterative and/or concatenated channel coding, in this context we cite the following references [136], [137], [38],[138] and [39].

After deriving the STVQ algorithm and showing the dependency of the algorithm's performance on the parameter η, we introduced the FSTVQ codebook search approach. The development of the new approach was motivated and based on several factors. The first one is the availability of soft distortion-related information delivered by the STVQ algorithm. The second factor is related to the rate distortion theory. As an introduction to FSTVQ, we showed the relationship between the basic theory behind Blahut algorithm, and the measures used and delivered by the STVQ algorithm. This comparison justified introducing the soft distortion measure, which led to the FSTVQ codeword update equations. The last factor is the fuzzy part of the new approach, which was introduced based on the dependency of the performance of the STVQ algorithm on the parameter η. In general, fuzzy coding has been shown to provide reliable results with deterministic annealing, which we introduced

earlier in chapter 2. It is known that the performance of the LBG codebook search algorithm is highly dependent on the initialization step as well as the length of the training sequence. However, searching for the proper long training sequence as well as the proper initial codebook is not always feasible due to practical nonstationarity conditions. Simulation results showed that the FSTVQ codebook search approach outperforms the LBG algorithm by providing lower distortion trellis codebook configurations. The improved performance is achieved while being significantly less sensitive to initial codebooks using relatively short training sequences.

The LBG algorithm suffers from the same deficiencies when applied with channel-optimized TVQ. Although, the sensitivity of the LBG algorithm to initial conditions becomes less noticeable as the level of channel noise increases, the algorithm is still dependent on initial conditions over a practical range of channel noise. In chapter 5 we extended the derivation of the FSTVQ algorithm to the case of channel-optimized TVQ. Simulation results showed the advantage of using channel-optimized FSTVQ as opposed to the channel-optimized LBG algorithm in providing lower-distortion trellis codebooks. In addition to that, the computed minimum distortion codebooks showed reasonable robustness under channel-mismatch conditions. We further showed the effect of different design channel BER values on the generated channel optimized codebooks in providing different robustness levels under channel-mismatch conditions. Besides that, we showed the features and differences between channel-optimized TVQ systems and tandem source and channel coding. It was shown that we can achieve the same BER levels achieved by tandem coding systems by just using channel-optimized compression, which reduces system complexity and processing delay.

The last part of this thesis considered introducing soft joint source/channel decoding/detection. The studied system involved transmitting k-DPCM compressed images over memoryless Gaussian channel with convolutional channel coding and MAP decoding. The redundancy level at the source encoder output is modeled by a 1st-order Markov model, which is used in a soft MAP-decoding metric as a source-optimized channel decoding approach. Several simulation cases emphasized the advantage of using higher redundancy levels with the JSC decoding metric, particularly during "bad" channel conditions. Improved performance was achieved after matching the alphabets of the source encoder output to the channel encoder input. For this purpose, we used dual-k convolutional codes matched to the k-bit DPCM output symbols. Furthermore, in appendix E we provided a comparison between sequence-MAP, instantaneous-MAP and symbol-MAP detection as three possible choices for JSC detection. As applied with VQ-compressed images, simulation results showed the advantage of symbol-MAP detection in providing lower average distortion of reproduction. This is due to the fact the that the symbol-MAP

approach minimizes the probability of incorrectly detected symbols, which improves the chances of reproducing correct VQ codevectors. It is worth mentioning at this point that several studies have considered using the soft information delivered by the symbol-MAP channel decoding/detection algorithm for a soft input source decoding approach [18], [19] and [20].

Appendix A

Background on Selected Sources

In this appendix we introduce the different sources, which are used mainly to test the different compression systems proposed in this thesis. Moreover, we present the theoretical distortion rate results for these sources of interest.

A.1 The Memoryless Gaussian Source

Memoryless sources deliver sequences of independent random variables. The Gaussian source is a frequently used source for standard comparisons. The pdf for a zero-mean Gaussian random variable of σ_y^2 variance is given by [124]

$$p_G(y) = \frac{1}{\sqrt{2\pi}\sigma_y} \exp\left\{\frac{-y^2}{2\sigma_y^2}\right\} \tag{A.1}$$

The uncertainty measure for continuous-amplitude sources is quantified by the differential entropy. The differential entropy of the Gaussian source is

$$h_G(y) = \frac{1}{2}\log_2\left\{2\pi e\sigma_y^2\right\} \quad \text{bits/sample} \tag{A.2}$$

It is clear from Eq. (A.2) that the amount of information of a given source is proportional to the variance of its output samples around their statistical average. It is known that the Gaussian source has the maximal differential entropy amongst all other sources. In other words, the output of a Gaussian source contains the maximal

amount of information per unit variance. In order to measure the relative amount of information per unit variance of different sources, we introduce the quantity of *entropy power* [41]

$$\mathcal{V} = \frac{1}{2\pi e} 2^{2h(y)} \tag{A.3}$$

Note that the entropy power of a Gaussian source $\mathcal{V}_G = \sigma_y^2$.

A.2 Sources with memory

The samples of the output sequence of sources with memory are correlated. The statistical dependencey between successive samples results in reducing the amount of uncertainty (information) at the source output. This means that the entropy rate of sources with memory is less than that of memoryless sources. Consequently, the source memory is usually exploited in the design of coding systems to provide the expected lower rates of compression. Assuming that the source output sequence is statistically predictable, it is mathematically correct to represent the source memory by a Markov model. In accordance with this hypothesis, there exists a white Gaussian process, which represents the difference between the source output sequence and a linear-filtered version of the same sequence. A mathematical treatment of this point is presented in [139]. The variance of the white Gaussian process represents the *minimum prediction error variance*, and it is given by

$$\eta_y^2 = \exp\left\{\frac{1}{2\pi}\int_{-\pi}^{\pi} \ln S_{yy}(e^{jw})dw\right\} \tag{A.4}$$

where $S_{yy}(e^{jw})$ represents the source power spectral density. It can be shown that the entropy power of sources with memory is equal to the minimum prediction error variance [41]. This equivalencey is intuitively justified, since as the sequence becomes less predictable (high η_y^2), its uncertainty increases. One important measure associated with sources with memory is the spectral flatness γ_y^2, and it is given by [140, 41]

$$\gamma_y^2 = \frac{\eta_y^2}{\sigma_y^2} \tag{A.5}$$

where the variance σ_y^2 is related to the power spectral density of the random variable y through

$$\sigma_y^2 = \frac{1}{2\pi}\int_{-\pi}^{\pi} S_{yy}(e^{jw})dw \tag{A.6}$$

For the special case of a white process, $S_{yy}(e^{jw}) = \sigma_y^2$, thus, $\gamma_y^2 = 1$. Otherwise

$$0 \leq \gamma_y^2 \leq 1.$$

In summary, we define the entropy power of sources with memory in terms of the spectral flatness of the source output

$$\mathcal{V} = \gamma_y^2 \cdot \sigma_y^2 = \eta_y^2 \tag{A.7}$$

It is clear from Eq. (A.7) that higher degrees of nonflatness (low γ_y^2), which is a feature of predictable sequence, reduces the amount of information inherent in the source samples.

A.3 Distortion-Rate Results

In this section we present the theoretical distortion-rate functions associated with the sources introduced earlier. These distortion-rate limits form lower bounds for practical attainable distortion values at the corresponding compression rates. As explained in appendix C, the objective distortion of interest in this thesis is the mean-squared-error (MSE) distortion. In order to introduce these theoretical limits, which assume infinite-length block processing, we show how the distortion rate function is defined in the context of finite-length block compression. In particular, the L-block distortion-rate function, which is associated with L-dimensional vector processing, is defined as [41]

$$D_L(R) = \min_{p(Y|\hat{Y}) \in \Upsilon} \left\{ E[d_L(Y, \hat{Y})] \right\} \tag{A.8}$$

where

$$\left.\begin{aligned} \Upsilon &= \left\{ p(Y|\hat{Y}) : I_L(Y, \hat{Y}) \leq R \right\} \\ d_L(Y, \hat{Y}) &= \frac{1}{L} \sum_{l=1}^{L} [Y_l - \hat{Y}_l]^2 \end{aligned}\right\} \tag{A.9}$$

and $\hat{Y}$ is an $L \times 1$ reproduction codeword. Then the distortion-rate function is given by

$$D(R) = \lim_{L \to \infty} D_L(R) \tag{A.10}$$

It is shown that the MSE distortion-rate function for memoryless, continuous-amplitude sources with zero-mean and σ_y^2 variance is upper-bounded by the distortion-rate function of a Gaussian source, which is given by [2]

$$D_G(R) = 2^{-2R} \sigma_y^2 \tag{A.11}$$

where R is given in bits/sample. On the other hand, any MSE distortion-rate function is lower bounded by the Shannon lower bound [124]

$$D_{\mathcal{L}}(R) = \frac{1}{2\pi e} 2^{-2[R - h(y)]} \tag{A.12}$$

Therefore, the MSE distortion rate function of any memoryless, continuous source is bounded by

$$D_{\mathcal{L}}(R) \leq D(R) \leq D_G(R) \tag{A.13}$$

To have a sense of the deviation between the lower and upper bounds of Eq. (A.13), we find a close-form expression of the difference $D_G(R) - D_{\mathcal{L}}(R)$. Using Eq. (A.2), we express the Shannon lower bound, of Eq. (A.12), in dB per unit variance as follows

$$10 \log_{10} D_{\mathcal{L}}(R) = -6R - 6[h_G(y) - h(y)] \tag{A.14}$$

It follows that

$$10 \log_{10} \frac{D_G(R)}{D_{\mathcal{L}}(R)} = 6[h_G(y) - h(y)] \quad \text{dB per unit } \sigma_y^2 \tag{A.15}$$

In theorem 4.5.3 in [2], Berger presents a parametric model for the rate-distortion function for the nth-order Gauss-Markov source. This parametric form is used in [141] and more recently in [142] to compute the different theoretical rate-distortion functions for the class of wide sense Gauss-Markov sources. As an example, we implement the theoretical development in [142] to compute the distortion rate function for different AR(1) and AR(2) sources as shown in Figs. A.1 and A.2. An interesting treatment of the computation of the rate distortion function for correlated sources is presented in [143]. The development in [143] is based on transforming the correlated source into a set of statistically independent coefficients. This factorization aided in expressing the rate distortion function in terms of the source covariance matrix. This particular development is used to compute the rate distortion function of the AR(2) source ($a_1 = 1.515$, $a_2 = -0.752$), which is shown in Fig. A.3. Nonetheless, for the range of small distortion, the distortion-rate function of a Gaussian source with memory is given by [41]

$$D_G|_{\text{memory}} = \gamma_y^2 D_G|_{\text{memoryless}} \tag{A.16}$$

where γ_y^2 is the source spectral flatness defined in Eq. (A.5). The definition of Eq. (A.16) is valid in the region

$$S_{yy}(e^{jw}) \geq D_G(R)$$

Once again, the distortion-rate function of Gaussian sources with memory serve as an upper bound for the distortion-rate function of other sources with memory. Consider the the first-order autoregressive (AR) Gaussian (thus Markov) source

$$y_n = z_n + \rho\, y_{n-1}$$

where z_n is a white Gaussian process of variance η_y^2 given in Eq. (A.4). The spectral flatness of the AR(1) source is given by [69]

$$\gamma_y^2 = 1 - \rho^2$$

Accordingly, the distortion-rate function of a 1st-order Gauss-Markov source is given by

$$D_G = 2^{-2R}(1-\rho^2)\sigma_y^2 \tag{A.17}$$

Eq. (A.17) is valid in the range of small distortion, which is restricted by the following condition [41]

$$R \geq \log_2(1+\rho) \tag{A.18}$$

Next we consider the second order Gauss Markov source defined as

$$y_n = z_n + a_1 y_{n-1} + a_2 y_{n-2} \tag{A.19}$$

The power spectral density of this source is given by [41]

$$S_{yy}(e^{jw}) = \frac{\sigma_y^2}{1 + a_1^2 + a_2^2 - 2a_1(1-a_2)\cos\omega - 2a_2 cos2\omega} \tag{A.20}$$

It is easy to show that the spectral flatness of this source is given by (see problem 2.15 in [41])

$$\gamma_y^2 = \frac{(1+a_2)(1-a_1-a_2)(1+a_1-a_2)}{(1-a_2)}$$

This definition can be used to solve for the $D(R)$ values in the range where $S_{yy}(e^{jw}) \geq D_G(R)$ according to Eq. (4.18).

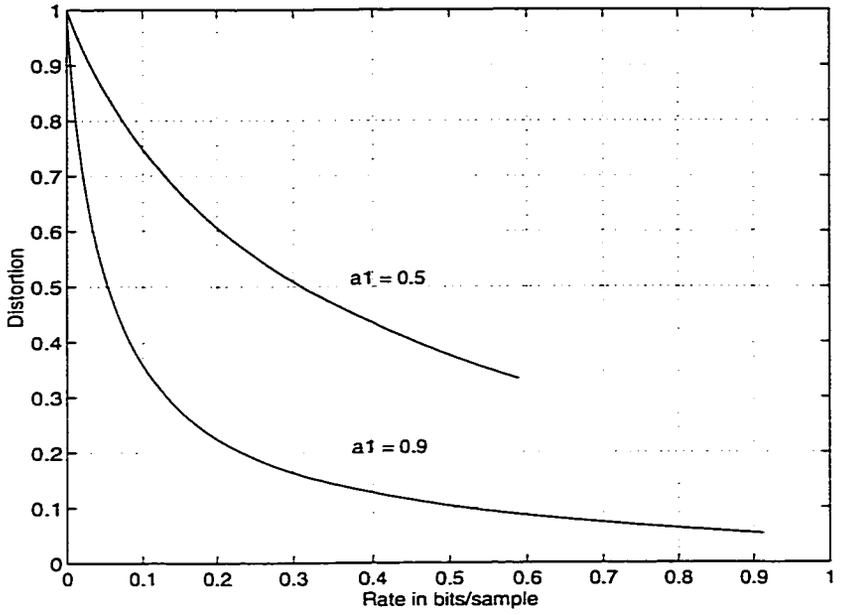

Figure A.1: The distortion rate functions for the AR(1) source. The distortion values are per unit source variance.

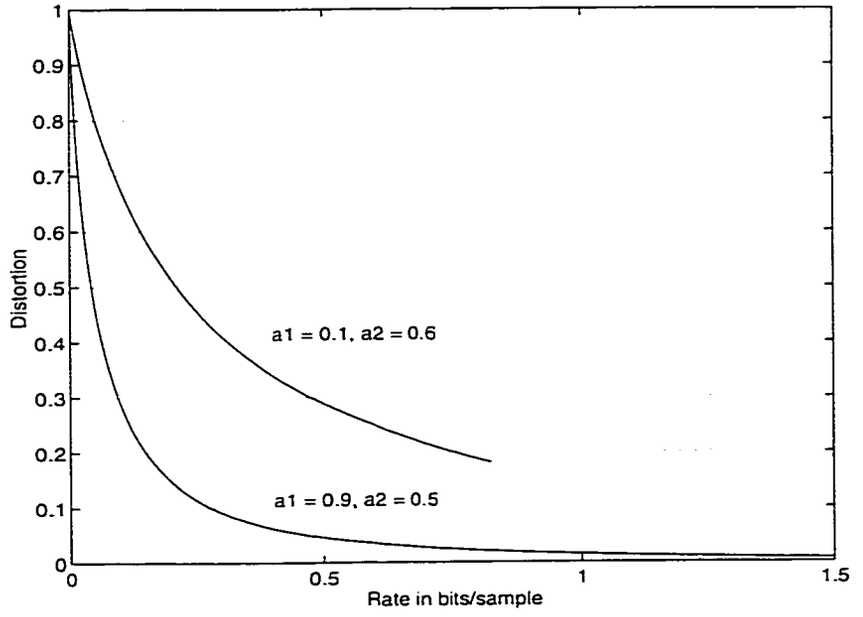

Figure A.2: The distortion rate function for the AR(2) source. The distortion values are per unit source variance.

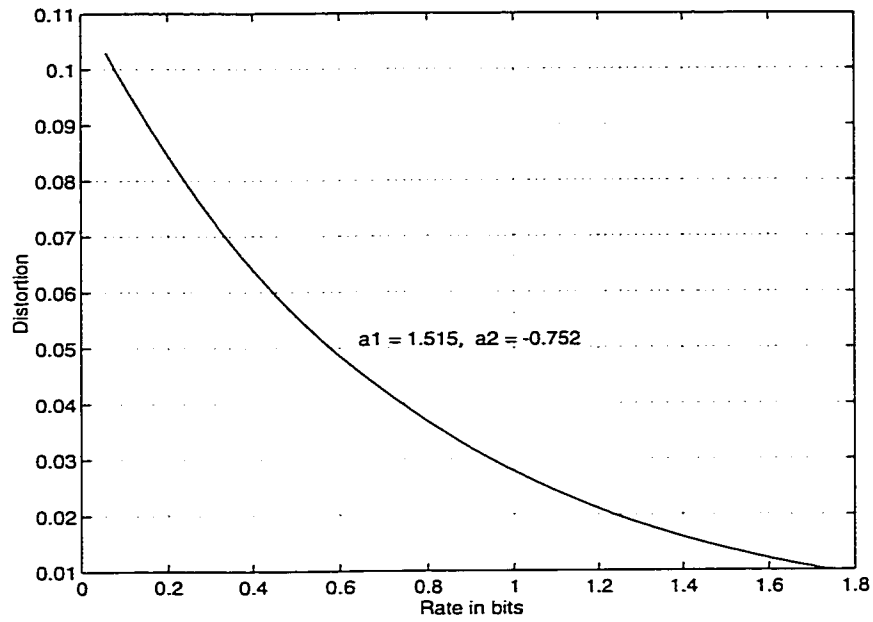

Figure A.3: The distortion rate function for the AR(2) source used in the thesis simulations. The distortion values are per unit source variance.

Appendix B

Classification of Residual Redundancy

The residual redundancy at the source encoder output is usually characterized by the amount of nonuniformity in the alphabet distribution as well as the level of correlation [27]. In order to explain the two folds of redundancy we introduce the concept of entropy rate, which represents the entropy of the whole sequence at the encoder output [144]

$$\begin{aligned} H_\infty(x) &= \lim_{N\to\infty} \frac{1}{N} H(X^N) \\ &= \lim_{N\to\infty} H(x_N|x_{N-1},\ldots,x_1) \end{aligned} \tag{B.1}$$

It can be easily shown that conditioning reduces the amount of information, therefore

$$\begin{aligned} H_\infty(x) &\leq \frac{\sum_{n=1}^{N} H(x_n)}{N} \\ &\leq H(x) \end{aligned} \tag{B.2}$$

Consequently, we define the part of redundancy that is related to the level of memory (correlation) as

$$\rho_M = H(x) - H_\infty(x) \tag{B.3}$$

On the other hand, the nonuniformity part of redundancy is defined as the difference between the number of bits $B = \log_2 J$ required to represent the output alphabet

and the symbol entropy

$$\rho_D = B - H(x) \tag{B.4}$$

Then, the total redundancy at the source encoder output is defined as [27]

$$\rho_T = \rho_M + \rho_D \tag{B.5}$$

Appendix C

Performance Measures

In the context of data compression, the performance measure of interest in this thesis research is the mean squared distortion of reproduction. Our interest in this measure is attributed to the universality of this distortion measure in real life application, as well as to its relevance to perceptual measures. To start with, this measure represents a special case of the general weighted quadratic distortion measure, which is a contender distortion measure in speech compression applications [49], [53] and [145]. Furthermore, the authors in [146] cite several references that prove the pertinence of the MSE measure in image and video applications. In particular, they report the correlation between minimizing the MSE measure and the improved perceptual quality of JPEG2000-based compressed images.

The overall signal-to-quantization noise ratio (SNR) is used as a performance measure through out this thesis. This quantity measures the amount of average distortion of reproduction per unit-variance of a source output sequence. It is given by

$$\text{SNR} = 10 \log_{10} \frac{\sigma_y^2}{E\{d(Y, \hat{Y})\}} \; dB \tag{C.1}$$

where

$$d(Y, \hat{Y}) = \frac{1}{L} \sum_{l=1}^{L} [\mathbf{Y}_l - \hat{Y}_l]^2$$

For the special case of image compression, the performance is measured using the peak signal to noise ratio (PSNR) [33]

$$\text{PSNR} = 10 \log_{10} \frac{\sum_{l=1}^{L} Y_{pl}^2}{E\{d(Y, \hat{Y})\}} \; dB \tag{C.2}$$

where Y_{pl} is the highest intensity value used to represent a pixel. For example $Y_{pl} = 255$ in the case where each pixel is represented using 8-bit integers.

Appendix D

Source Correlation and the STVQ Algorithm

In this appendix we consider proving that the correlation between the different source output vectors has no weight on the search for the minimum distortion path over the compression trellis. To show this point we use the development of the STVQ algorithm. Assuming at first that the source output vectors are correlated, we write the joint probability variable defined in Eq. 4.17 as

$$\lambda_n^i(m) = \alpha_n^i(m) \cdot \beta_n^i(m) \tag{D.1}$$

where

$$\left.\begin{aligned} \alpha_n^i(m) &= \Pr\{x_n = i, s_n = m, Y_1^n\} \\ \beta_n^i(m) &= \Pr\{Y_{n+1}^N | x_n = i, s_n = m, Y_1^n\} \end{aligned}\right\} \tag{D.2}$$

Recall that $\beta_n^i(m)$ is independent of Y_1^n in the memoryless version of the BCJR algorithm.

In the following we recursively solve for the forward variable $\alpha_n^i(m)$. Note that

$$\alpha_n^i(m) = \sum_{j=0}^{I-1} \sum_{\acute{m}=0}^{M-1} \Pr\{x_n = i, x_{n-1} = j, s_n = m, s_{n-1} = \acute{m}, Y_1^n\} \tag{D.3}$$

It follows that

$$\begin{aligned}\alpha_n^i(m) &= \sum_{j=0}^{I-1}\sum_{\acute{m}=0}^{M-1} \Pr\{x_n = i, x_{n-1} = j, s_n = m, s_{n-1} = \acute{m}, y_n, Y_1^{n-1}\} \\ &= \sum_{j=0}^{I-1}\sum_{\acute{m}=0}^{M-1} \Pr\{x_{n-1} = j, s_{n-1} = \acute{m}, Y_1^{n-1}\} \\ &\cdot \Pr\{x_n = i, s_n = m, y_n | x_{n-1} = j, s_{n-1} = \acute{m}, Y_1^{n-1}\} \end{aligned} \tag{D.4}$$

Let $\mathcal{T}$ be the probability variable in the third line of Eq. (D.4), then, we simplify this variable as

$$\begin{aligned}\mathcal{T} &= \Pr\{x_n | x_{n-1} = j, s_{n-1} = \acute{m}, Y_1^{n-1}\} \cdot \Pr\{s_n = m | x_{n-1} = j, s_{n-1} = \acute{m}, Y_1^{n-1}\} \\ &\cdot \Pr\{y_n | x_{n-1} = j, s_{n-1} = \acute{m}, x_n = i, s_n = m, Y_1^{n-1}\} \end{aligned} \tag{D.5}$$

In anticipation of delivering a maximal information sequence (maximum entropy sequence), we consider an uncorrelated and equally-probable source encoder output symbols (ideal compression). This unbiased constraint, which is implicitly imposed by all trellis waveform coding algorithms, ensures making use of the available trellis encoding power. Thus, $\Pr\{x_n | x_{n-1} = j, s_{n-1} = \acute{m}, Y_1^{n-1}\} = \Pr\{x_n\} = 1/I$, which is a constant that has no weight in the process of computing the different variables. Furthermore, $\Pr\{s_n = m | x_{n-1} = j, s_{n-1} = \acute{m}, Y_1^{n-1}\}$ has a zero or one value, which specifies whether we are following the right trellis branch or not. Thus, as in section 4.2, we define $S_b^j(m)$ as the state at which we arrive if we go backward from state $s_n = m$ following the branch $x_{n-1} = j$. Regarding the third term in Eq. (D.5), encoding y_n is controlled only by the state at time n, the branch index at time n, and due to the considered source memory, the previous source output symbols. Accordingly, we the intermediate variable $\mathcal{T}$ of Eq. (D.5) reduces to the conditional probability variable

$$\gamma_n^i(m) = \Pr\{y_n | x_n = i, s_n = m, Y_1^{n-1}\} \tag{D.6}$$

Based on the previous argument, Eq. (D.4) reduces to

$$\alpha_n^i(m) = \gamma_n^i(m) \sum_{j=0}^{I-1} \alpha_{n-1}^j(S_b^j(m)) \tag{D.7}$$

At this point the problem at hand is to find a close form solution to Eq. (D.6). Following an approach similar to MAP-decoding over channels with memory, which is treated in [147], [148], and [149], we found a recursion to solve for $\gamma_n^i(m)$ in Eq. (D.6). Simulation results of the derived equations delivered unexpected and rather

incorrect outcomes. The problem was inherent in the definition of the distortion metric that we are trying to minimize. To explain the problem, we first replace x_n and s_n Eq. (D.6) by $c_n^{m,i}$, which stands for the codeword that colors the branch $x_n = i$ out of state $s_n = m$. Accordingly,

$$\gamma_n^i(m) = \Pr\{y_n | c_n^{m,i}, Y_1^{n-1}\} \tag{D.8}$$

or equivalently

$$\begin{aligned} \gamma_n^i(m) &= \frac{\Pr\{y_n, Y_1^{n-1} | c_n^{m,i}\}}{\Pr\{Y_1^{n-1} | c_n^{m,i}\}} \\ &= \frac{\Pr\{y_n | c_n^{m,i}\} \Pr\{Y_1^{n-1} | y_n, c_n^{m,i}\}}{\Pr\{Y_1^{n-1} | c_n^{m,i}\}} \end{aligned} \tag{D.9}$$

Due to the source correlation (memory), the sequence Y_1^{n-1} provides statistical information about the vector y_n. Therefore, ignoring the past $n-1$ source output vectors in the definition of the different probability variables, which appear in Eq. (D.9), seems rather impetuous. However, keep in mind that the ultimate goal is to minimize the cumulative distortion measure, which represents the sum of the "distances" between the source output vectors and the constituent reproduction codevectors at each time instant. Based on this objective we shall be interested in finding the best codeword that is fit to represent a source output vector for each time instant. This basically culminated into defining $\Pr\{y_n | c_n^{m,i}\}$ in Eq. (4.11). Thus, as far as searching for the minimum distortion path is concerned, $\Pr\{Y_1^{n-1} | y_n, c_n^{m,i}\} = \Pr\{Y_1^{n-1} | y_n\}$, besides, $\Pr\{Y_1^{n-1} | c_n^{m,i}\} = \Pr\{Y_1^{n-1}\}$. Employing the results of this argument in Eq. (D.9) produces

$$\begin{aligned} \gamma_n^i(m) &= \Pr\{y_n | c_n^{m,i}\} \frac{\Pr\{Y_1^{n-1} | y_n\}}{\Pr\{Y_1^{n-1}\}} \\ &= \Pr\{y_n | c_n^{m,i}\} \frac{\Pr\{y_n | Y_1^{n-1}\}}{\Pr\{y_n\}} \end{aligned} \tag{D.10}$$

Clearly the contribution of the source memory to the computation of the variable $\gamma_n^i(m)$ is independent of the indices i, m. Consequently, it can be dropped out without affecting the performance of the STVQ algorithm. In conclusion, unlike the case of MAP-decoding over channels with memory, the source memory cannot be used over the trellis to minimize the overall distortion of reproduction. This is mainly attributed to the fact that redundancy in the channel decoding case is introduced before passing through the channel with memory. Thereby, having a clear statistical model, which describes the relation between the received sequence and the original codewords on the channel decoding trellis.

For the sake of completeness, we derive the recursion of the backward variable $\beta_n^i(m)$ defined in Eq. (D.2). We first write the backward variable as

$$\beta_{n-1}^j(\acute{m}) = \Pr\{Y_n^N | x_{n-1} = j, s_{n-1} = \acute{m}, Y_1^{n-1}\} \tag{D.11}$$

Then, we solve for $\beta_{n-1}^j(\acute{m})$ using the recursion

$$\begin{aligned}
\beta_{n-1}^j(\acute{m}) &= \sum_{i=0}^{I-1}\sum_{m=0}^{M-1} \Pr\{s_n = m, x_n = i, y_n, Y_{n+1}^N | x_{n-1} = j, s_{n-1} = \acute{m}, Y_1^{n-1}\} \\
&= \sum_{i=0}^{I-1}\sum_{m=0}^{M-1} \Pr\{s_n = m, x_n = i, y_n | x_{n-1} = j, s_{n-1} = \acute{m}, Y_1^{n-1}\} \\
&\quad \cdot \Pr\{Y_{n+1}^N | s_n = m, x_n = i, x_{n-1} = j, s_{n-1} = \acute{m}, Y_1^n\}
\end{aligned} \tag{D.12}$$

However, given the values of s_n, x_n, the trellis path is completely specified before s_{n+1}. Thus,

$$\Pr\{Y_{n+1}^N | s_n = m, x_n = i, x_{n-1} = j, s_{n-1} = \acute{m}, Y_1^n\} = \beta_n^i(m) \tag{D.13}$$

Furthermore, based on the constraint of ideal compression, the probability variable in line 2 of Eq. (D.12) can be simplified into

$$\Pr\{x_n = i\} \cdot \Pr\{s_n = m | x_{n-1} = j, s_{n-1} = \acute{m}\} \cdot \gamma_n^i(m)$$

with $\gamma_n^i(m)$ defined in Eq. (D.6) and eventually in Eq. (D.10). The variable $\Pr\{s_n = m | x_{n-1} = j, s_{n-1} = \acute{m}\}$ has a one/zero value, which specifies the state $s_n = S_f^j(\acute{m})$ at which we arrive if we leave state $s_{n-1} = \acute{m}$ along the branch $x_{n-1} = j$. Therefore,

$$\beta_{n-1}^j(\acute{m}) = \sum_{i=0}^{I-1} \beta_n^i(S_f^j(\acute{m})) \cdot \gamma_n^i(S_f^j(\acute{m})) \tag{D.14}$$

Appendix E

Source-Optimized Channel Detection

This appendix examines sending the information sequence over the communications channel without channel coding. In this system the residual redundancy at the source encoder output (lossy compression system) is used to enhance the reliability of detected symbols at the channel output. In their papers [18] and [35] the authors use this approach to design a JSC coding system for transmitting Markov sources over discrete memoryless as well as Markov channels. The authors use the statistics of a Markov source to develop two *Maximum A Posteriori* detectors, one minimizes the probability of received sequences in error, and the other minimizes the probability of received symbols in error. The latter, however, is instantaneous in the sense that it decides on the transmitted symbol at the same time the channel output is received. In this appendix, we compare through several cases the performance of sequence-MAP and instantaneous-MAP to the performance of symbol-MAP detection.

As shown in section 3.1.1, given an L-symbol channel output sequence, the symbol-MAP algorithm minimizes the probability of received symbols in error. As compared to sequence-MAP and instantaneous-MAP, symbol-MAP shows similar or improved bit error rate performance in several simulation conditions with hard as well as soft detection. Note that, maximizing the probability of correct symbols at the source decoder input reflects into maximizing the probability of correct reproduction codewords. We will show that employing symbol-MAP detection with lossy symbol-

based compression techniques provides lower average distortions of reproduction. In addition to a binary symmetric Markov source, the performance of the three detection approaches is tested via transmitting vector quatized (VQ) gray images. Furthermore, the effect of changing the correlation level, at the channel input, on the detection gain, is demonstrated in the simulations section. Improved system performance is expected by incorporating the design of the source coding system and the channel detection part. Several points regarding this system are addressed in [18], [19] and [20], which is generally referred to in literature as soft input source decoding.

E.1 Background and System Configuration

In this appendix we consider the transmission of first-order Markov sources over AWGN channels with spectral height $N_o/2$. The transmission is assumed binary, however, the channel input is assumed to be the output of a stationary discrete source $\{X_n\}_{n=1}^{\infty}$ with alphabet $\mathcal{A} \equiv \{0, 1, 2, \ldots, J-1\}$. The source statistics are characterized by the following transition probability matrix

$$\{P(x_n|x_{n-1})\} = \{P(X_n = x_n|X_{n-1} = x_{n-1})\} \quad x_n, x_{n-1} \in \mathcal{A}$$

Note that this source is a typical output of a practical source compression system. The set $\{\Pr\{x_n = i|x_{n-1} = j\}\}$ is estimated at the channel input using a training sequence. In [18], the authors consider the problem of optimal detection of Markov sources over discrete memoryless channels. Their approach is based on both sequence-MAP and instantaneous-MAP detection.

E.1.1 Sequence-MAP Detection

Similar to sequence-MAP decoding, the detector in this class decides on the most probable transmitted sequence of length L according to [18]

$$\hat{X}_1^L = \arg \max_{X_1^L \in \mathcal{A}^L} \Pr\{X_1^L|\hat{Y}_1^L\}$$

$$\hat{X}_1^L = \arg \max_{X_1^L \in \mathcal{A}^L} \left[\sum_{n=2}^{L} \log[Q(\hat{y}_n|x_n)P(x_n|x_{n-1})] + \log[Q(\hat{y}_1|x_1)P(x_1)]\right] \quad \text{(E.1)}$$

where $\hat{Y}_1^L = \{\hat{y}_1, \hat{y}_2, ..., \hat{y}_L\}$ is the channel output sequence, and $Q(\cdot|\cdot)$ is used to denote the channel distribution. The Viterbi algorithm is used to search for the

MAP sequence using the following metric

$$M_n = \max_{x_{n-1},x_n} \left\{ M_{n-1} + \log P(x_n|x_{n-1}) + \log \prod_{j=1}^{B} Q(\hat{y}_n^j|x_n^j) \right\} \tag{E.2}$$

where $B = \log_2 J$ bit/symbol. Consequently, to perform soft Viterbi decoding we substitute the Gaussian distribution for $Q(\hat{y}_n^j|x_n^j)$ in Eq. (E.2) to get

$$M_n = \max_{x_{n-1},x_n} \left\{ M_{n-1} + \frac{N_o}{E_b} \log P(x_n|x_{n-1}) + \sum_{j=1}^{B} \frac{2\hat{y}_n^j x_n^j}{\sqrt{E_b}} \right\} \tag{E.3}$$

E.1.2 Instantaneous-MAP Detection

The decision in this class is made based on present and past received channel symbols. This detection scheme maximizes the probability of detected symbols, however, unlike sequence-MAP detection, no delay is involved in this approach. The detection algorithm decides on the most probable transmitted symbol at time n according to the following equations, [18]

$$\begin{aligned} \hat{x}_n &= \arg\max_{x_n \in \mathcal{A}} \Pr\{\hat{Y}_1^n, x_n\} \\ &= \arg\max_{x_n \in \mathcal{A}} f^{(n)}(x_n) \qquad n = 1, 2, \ldots . \end{aligned} \tag{E.4}$$

where

$$\left.\begin{aligned} f^{(1)}(x_1) &= Q(\hat{y}_1|x_1)\, P(x_1) \\ f^{(n)}(x_n) &= Q(\hat{y}_n|x_n) \sum_{x_{n-1} \in \mathcal{A}} P(x_n|x_{n-1}) \cdot f^{(n-1)}(x_{n-1}) \quad n = 2, 3, \ldots . \end{aligned}\right\}$$

E.1.3 Symbol-MAP Detection

After receiving a sequence of channel output symbols, this algorithm decides on the transmitted symbols such that

$$\hat{x}_n = \arg\max_{x_n \in \mathcal{A}} \Pr\{x_n|\hat{Y}_1^L\} \tag{E.5}$$

Thus, the algorithm minimizes the probability of detected symbols in error. The procedure for this algorithm is completely described in section 3.1.1.

E.2 Simulation Results

In the sequel, we present the performance results of the different detection algorithms mentioned earlier. The transmission is BPSK over AWGN channels with bit-error-rate ϵ. In all figures, the bit-error-rate (Pe) performance curves are plotted along with the no MAP-detection line, which represents the case where Pe $= \epsilon$.

E.2.1 Binary Symmetric Markov Source

In this case the source is binary symmetric Markov (BSMS) with the following transition probabilities

$$P(x_k|x_{k-1}) = \begin{cases} p & x_k \neq x_{k-1} \\ 1-p & x_k = x_{k-1} \end{cases}$$

The transmission is over a binary symmetric channel; i.e., hard detection. The simulation results for the cases with $p = 0.05$ and $p = 0.01$ are shown in Figs. E.1 and E.2, respectively. In these figures we demonstrate the close performance of symbol-MAP detection to sequence-MAP detection. Note also that higher detection gains are achieved with increased correlation levels (smaller p) at the channel input. For both cases of sequence-MAP and instantaneous-MAP detection, the authors in [18] show that the optimum detected sequence is the channel output sequence whenever ϵ is below certain values denoted as $\epsilon_{sequence}$ and $\epsilon_{istantaneous}$. This system behavior is clear in Figs. E.1 and E.2, where the curves merge into the "w/o MAP" line at different ϵ. However, the diverging point happens at smaller values of ϵ in Fig. E.2, where the algorithms have more correlation to enhance their detection capabilities at lower channel BER values.

E.2.2 Vector Quantized Source

The residual redundancy (correlation) at the output of a vector quantizer is used, in this case, by the detection algorithms to reduce the impact of channel errors. The source is an image represented by 8 bpp. A 16-dimensional VQ is used with a codebook of cardinality 4. Therefore, the source encoder produces 2-bit information symbols at a rate of 0.125 bpp. A training image is used to estimate the transition probabilities $\{P(x_k|x_{k-1})\}_{x_k \in \{0,1,2,3\}}$ at the channel input. Furthermore, the entropy of the source encoder output for the same training image equals 1.935 bps, which results in 0.065 bit of residual redundancy in each symbol. At this compression rate and for the used training image, the error-free PSNR is 22.55 dB. In Fig. E.3 we

Table E.1: The achieved PSNR values in dB for the reproduced transmitted image as a function of channel BER ϵ. The detection is soft, and the error-free PSNR = 22.55 dB.

	PSNR (dB)		
ϵ	w/o MAP	sym-MAP	seq-MAP
0.4	10.305	13.4571	11.49
0.1	10.614.	20.182	18.123
0.06	10.685	21.080	18.280
0.01	10.779	22.300	18.393
0.001	10.773	22.544	18.413

present the performance of the different detection algorithms. Note that the effect of soft detection in achieving higher error protection is clear in this figure.

Although the bit-error-rate performance of symbol-MAP detection is comparable to that of sequence-MAP, a symbol-MAP detector functions to provide the source decoder with sequences of more reliable codeword indices. This in turn enhances the quality of the reproduced sequence of information at the source encoder output. The significance of using symbol-MAP with VQ is clear in Table E.1, where we demonstrate higher PSNR values as compared to w/o MAP as well as sequence-MAP.

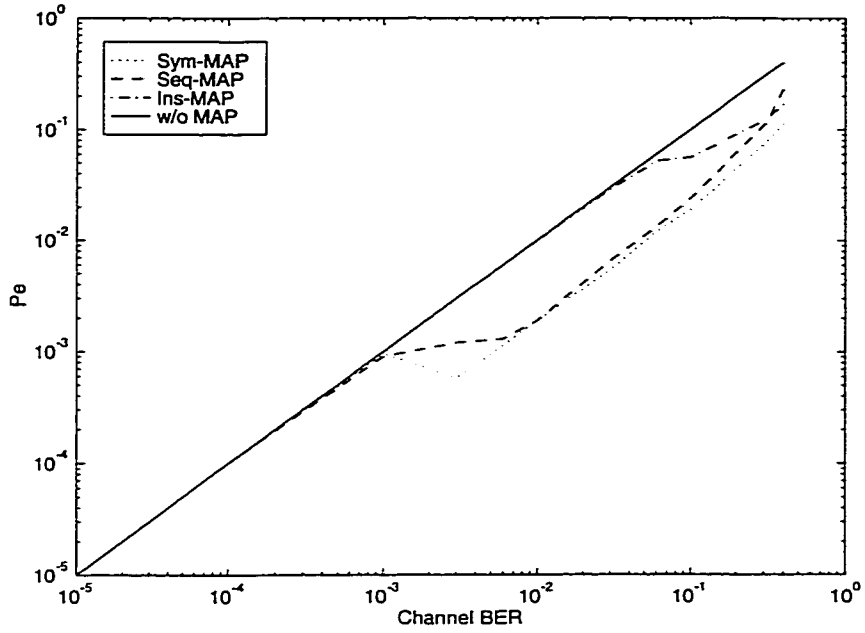

Figure E.1: The detection performance of the different algorithms with a BSMS of $p = .05$.

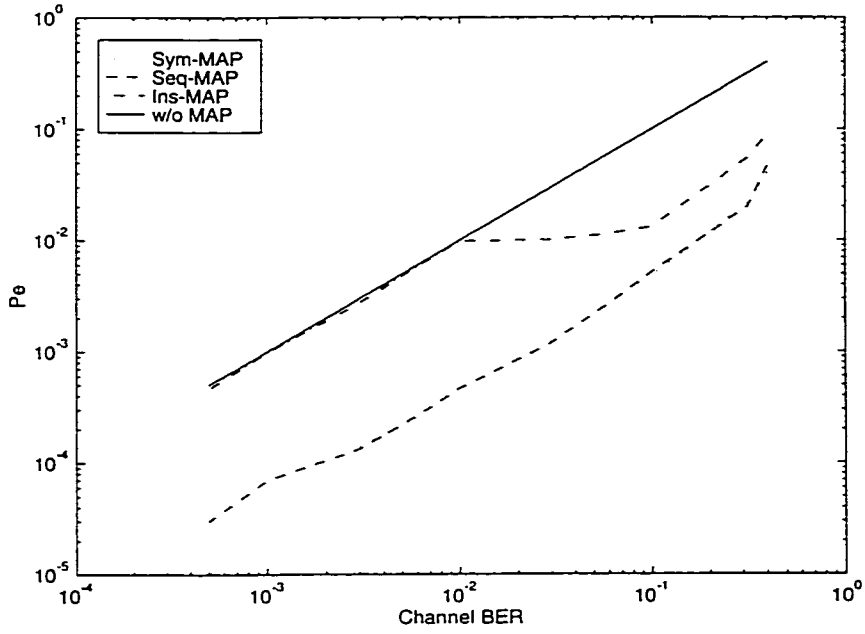

Figure E.2: The detection performance of the different algorithms with a BSMS of $p = .01$.

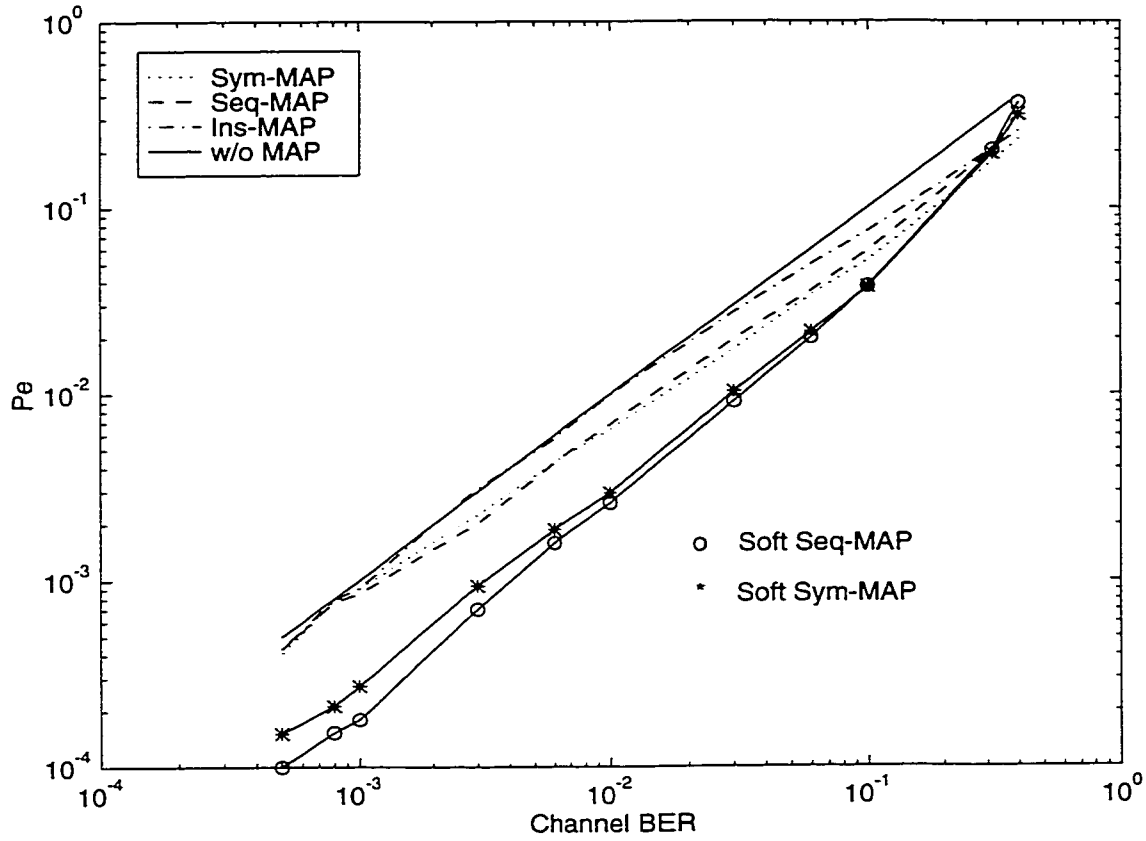

Figure E.3: The detection performance of the different algorithms with a VQ-compressed image.

Bibliography

[1] Joachim Hagenauer. "Source-Controlled Channel Decoding". *IEEE Trans. on Commun.*, 43(9), 1995.

[2] T. Berger. *Rate Distortion Theory, A Mathematical Basis For Data Compression.* Englewood Cliffs, NJ: Prentice-Hall, 1971.

[3] R. M. Gray. "Vector Quantization". *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 1:4–29, Apr. 1984.

[4] Harry L. Van Trees. *Detection, Estimation, and Modulation Theory*, volume I. John Wiley and Sons, New York, 1968.

[5] G. David Forney. "The Viterbi Algorithm". *Proceedings of the IEEE*, 61(3):268–278, Mar. 1973.

[6] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv. "Optimal decoding of linear codes for minimizing symbol error rate". *IEEE Transactions on Information Theory*, IT-20(2):284–287, Mar. 1974.

[7] R. G. Gallager. "Tree encoding for symmetric sources with a distortion measure". *IEEE Transactions on Information Theory*, IT-20:65–76, 1974.

[8] R. J. Dick, T. Berger, and F. Jelinek. "Tree encoding of Gaussian sources". *IEEE Transactions on Information Theory*, IT-20:332–336, May 1974.

[9] A. J. Viterbi and J. K. Omura. "Trellis encoding of memoryless discrete-time sources with a fidelity criterion". *IEEE Transactions on Information Theory*, IT-20:325–332, May 1974.

[10] R. M. Gray. "Time-invariant trellis encoding of ergodic discrete-time sources with a fidelity criterion". *IEEE Transactions on Information Theory*, IT-23(1):71–83, Jan. 1977.

[11] W. A. Pearlman. "Sliding-block and random source coding with constrained size reproduction alphabets". *IEEE Transactions on Communications*, COM-30:1859–1867, Aug. 1982.

[12] L. C. Stewart, R. M. Gray, and Y. Linde. "The design of trellis waveform coders". *IEEE Transactions on Communications*, COM-30(4):702–710, April 1982.

[13] W. A. Pearlman and A. Chekima. "Source coding bounds using quantizer reproduction levels". *IEEE Transactions on Information Theory*, IT-30:559–567, May 1984.

[14] M. W. Marcellin and T. R. Fisher. "Trellis coded quantization of memoryless and Gauss-Markov sources". *IEEE Transactions on Communications*, 38:82–93, Jan. 1990.

[15] E. Yair, K. zeger, and A. Gersho. "Competitive Learning and soft competition for vector quantization". *IEEE Transactions on Signal Processing*, 40(2):294–308, Feb. 1992.

[16] S. P. Lloyd. "Least squares quantization in PCM". *IEEE Transactions on Information Theory*, IT-28:129–137, Mar. 1982.

[17] R. Blahut. "Computation of channel capacity and rate distortion functions". *IEEE Transactions on Information Theory*, IT-18:460–473, 1972.

[18] Nam Phamdo and Nariman Farvardin. "Optimal detection of discrete Markov sources over discrete memoryless channels-applications to combined source-channel coding". *IEEE Transactions on Information Theory*, 40(1), Jan. 1994.

[19] M. Skoglund and P. Hedelinn. "Vector quantization over a noisy channel using soft decision decoding". In *Proceedings ICASSP-94 (IEEE International Conference on Acoustics, Speech and Signal Processing)*, volume 5, pages 605–608, April 1994.

[20] H. Skinnemoen and A. Perkis. "Efficient vector quantization of LPC parameters for noisy channel". In *Proceedings ICASSP-94 (IEEE International Conference on Acoustics, Speech and Signal Processing)*, volume 1, pages 497–500, April 1994.

[21] James G. Dunham and Robert M. Gary. "Joint source and noisy channel trellis encoding". *IEEE Transactions on Information Theory*, IT-27(4):516–519, July 1981.

[22] E. Ayanoglu and R. M. Gray. "The design of joint source and channel trellis waveform coders". *IEEE Transactions on Information Theory*, IT-33(6):855–865, Nov. 1987.

[23] N. Farvardin and V. Vaishampayan. "Optimal quantizer design for noisy channels: An approach to combined source channel coding". *IEEE Transactions on Information Theory*, pages 827–838, Nov. 1987.

[24] N. Farvardin. "A study of vector quantization for noisy channels". *IEEE Transactions on Information Theory*, 36(4):799–809, July 1990.

[25] Nariman Farvardin and Vinay Vaishampayan. "On the performance and Complexity of channel-optimized Vector quantizers". *IEEE Transactions on Information Theory*, 37(1):155–160, Jan. 1991.

[26] Nam Phamdo, Nariman Farvardin, and Takehiro Moriya. "A Unified approach to tree-structured and multistage vector quantization for noisy channels". *IEEE Transactions on Information Theory*, 39(3):835–850, May 1993.

[27] Nam Phamdo, Fady Alajaji, and Nariman Farvardin. "Quantization of memoryless and Gauss-Markov sources over binary Markov channels". *IEEE Transactions on Communications*, 45(6):668–675, June 1997.

[28] J. L. Massey. "Joint source and channel coding". In J. K. Skwirzynski, editor, *Communication Systems and Random Process Theory.* Ed. The Netherlands: Sijthoff and Nordhoff, 1978.

[29] M. E. Helman. "On using natural redundancy for error detection". *IEEE Transactions on Communications*, COM-22:1690–1693, Oct. 1974.

[30] M. E. Hellman. "Convolutional source encoding". *IEEE Transactions on Information Theory*, 21:651–656, 1975.

[31] K. N. Ngan and R. Steele. "Enhancement of PCM and DPCM images corrupted by transmission errors". *IEEE Transactions on Communications*, COM-30:257–269, Jan. 1982.

[32] R. Steele and D. J. Goodman. " Detection and selective smoothing of transmission errors in linear PCM". *BSTJ*, 56:399– 409, Mar. 1977.

[33] Jerry D. Gibson Khalid Sayood, Fuling Liu. "A constrained joint source/channel coder design". *IEEE Journal on Selected Areas in Communications*, 12(9):1584–1593, Dec. 1994.

[34] Khalid Sayood and J. C. Borkenhagen. "Use of redundancy in the design of joint source/channel coders". *IEEE Transactions on Communications*, 39(6), June 1991.

[35] Fady Alajagi, Nam Phamdo, Nariman Farvardin, and Thomas E. Fuja. "Detection of binary Markov sources over channels with additive Markov noise". *IEEE Transactions on Information Theory*, 42(1):230–239, Jan. 1996.

[36] C. Berrou, A. Glavieux, and P. Thitimajshima. "Near Shannon limit error-correction coding and decoding: Turbo codes". In *Proceedings ICC-93*, pages 1064–1070, Geneva, May 1993.

[37] S. Benedetto and G. Montorsi. "Unveiling turbo codes: Some results on parallel concatenated coding schemes". *IEEE Transactions on Information Theory*, 42(2):409–428, Mar. 1996.

[38] J. Hagenauer, E. Offer, and L. Papke. "Iterative decoding of binary block and convolutional codes". *IEEE Transactions on Information Theory*, 42(2):429–445, Mar. 1996.

[39] B. Sklar. "A primer on turbo code concepts". *IEEE Communications Magazine*, pages 94–102, Dec. 1997.

[40] C. E. Shannon. "Coding theorems for a discrete source with a fidelity criterion". *IRE Conv. Rec.*, 7:142–163, 1959.

[41] N. S. Jayant and Peter Noll. *Digital Coding of Waveforms, principles and Applications to Speech and Video.* Englewood Cliffs, NJ: Prentice-Hall, Inc., 1984.

[42] R. M. Gray and D. L. Neuhoff. "Quantization" . *IEEE Transactions on Information Theory*, 44(6):2325–2434, Oct. 1998.

[43] Sergio Verdu. "Fifty years of Shannon theory". *IEEE Transactions on Information Theory*, 44(6):2057–2078, Oct. 1998.

[44] R. E. Blahut. *Principles and practice of information theory.* Addison-Wesley, Massachusetts, 1987.

[45] Robert G. Gallager. *Information Theory and Reliable Communication.* John Wiley and Sons, Inc., New York, 1968.

[46] C. E. Shannon. "A mathematical theory of communication". *Bell Syst. Tech. J.*, 27:379–423 and 623–656, Oct 1948.

[47] T. Berger and J. D. Gibson. "Lossy source coding". *IEEE Transactions on Information Theory*, 44(6):2693–2723, Oct. 1998.

[48] A. Gersho. "On the structure of vector quantization". *IEEE Transactions on Information Theory*, IT-28:157–166, March 1982.

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

[50] Khalid Sayood. *Introduction to data compression.* Morgan Kaufmann Publishers, Inc., San Francisco, California, 1996.

[51] E. Ayanoğlu and R. M. Gray. "The design of predictive trellis waveform coders using the generalized Lloyd algorithm". *IEEE Transactions on Communications*, COM-34(11):1073–1080, Nov. 1986.

[52] M. J. Sabin and R. M. Gray. "Global convergence and empirical consistency of the generalized Lloyd algorithm ". *IEEE Transactions on Information Theory*, IT-32(2):148–155, Mar. 1986.

[53] R. M. Gray and E. D. Karnin. "Multiple local minima in vector quantizers". *IEEE Transactions on Information Theory*, IT-28:256–261, March 1982.

[54] K. Zeger, J. Vaisey, and A. Gersho. "Globally optimal vector quantizer design by stochastic relaxation". *IEEE Transactions on Signal Processing*, 40(2):310–322, Feb. 1992.

[55] U. Moller, M. Galicki, E. Baresova, and H. Witte. "An efficient vector quantizer providing globally optimal solutions". *IEEE Transactions on Signal Processing*, 46(9):2515–2529, Sep. 1998.

[56] Kenneth Rose. "Deterministic annealing for clustering, compression, classification, regression, and related optimization problems". *Proceedings of the IEEE*, 86(11):2210–2239, Nov. 1998.

[57] K. Rose, E. Gurewitz, and G. C. Fox. "Vector quantization by deterministic annealing". *IEEE Transactions on Information Theory*, 38(4):1249–1257, July 1992.

[58] Abbas A. Gamal, Lane A. Hemachandar, I. Sherling, and V. K. Wei. "Using simulated annealing to design good codes". *IEEE Transactions on Information Theory*, IT-33(1):116–123, Jan. 1987.

[59] J. R. B. Marca and N. S. Jayant. " An algorithm for assigning binary indices to the codevectors of a multidimensional quantizer ". In *Proceedings ICC-87 (IEEE International Conference on Communications)*, pages 1128–1132, Seattle, WA, June 1987.

[60] L. T. Wille. "New binary covering codes obtained by simulated annealing". *IEEE Transactions on Information Theory*, 42(1):300–302, Nov. 1994.

[61] Jr. M. P. Vecchi S. Krikpatrick, C. D. Gelatt. "Optimization by Simulated Annealing". *Science*, 220:671–680, May 1983.

[62] M. N. Rosenbluth A. H. Teller N. Metropolis, A. W. Rosenbluth and E. Teller. "Equation of state calculations by fasting computing machines". *J. Chem. Phys.*, 21, June 1953.

[63] R. Durbin, R. Szeliski, and A. Yuille. "An analysis of the elastic net approach to the travelling salesman problem". *Neural Computation*, 1(3):348–358, Mar. 1989.

[64] K. Rose, E. Gurewitz, and G. C. Fox. "A deterministic annealing approach to clustering". *Pattern Recognition Letters*, 11(9):589–594, 1990.

[65] D. Geiger and F. Girosi. "Parallel and deterministic algorithms from MRFs: surface reconstruction". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:401–412, May 1991.

[66] F. Jelinek. "Tree encoding of memoryless time-discrete sources with a fidelity criterion". *IEEE Transactions on Information Theory*, IT-15(5), Sep. 1969.

[67] J. W. Mark. "Adaptive trellis encoding of discrete-time sources with a distortion measure". *IEEE Transactions on Communications*, COM-25(4):408–416, Apr. 1977.

[68] S. G. Wilson and D. W. Lytle. " Trellis Encoding of continuous-amplitude memoryless sources". *IEEE Transactions on Information Theory*, IT-23:404–409, May 1977.

[69] Heinz G. Fehn and Peter Noll. "Multipath search coding of stationary signals with applications to speech". *IEEE Transactions on Communications*, COM-30(4):687–701, Apr. 1982.

[70] J. B. Anderson and S. Mohan. "Sequential encoding algorithms: A survey and cost analysis". *IEEE Transactions on Communications*, COM-34:169–176, Feb. 1984.

[71] J. B. Anderson and J. B. Bodie. "Tree encoding of speech". *IEEE Transactions on Information Theory*, IT-21(4):379–387, 1975.

[72] T. R. Fisher, M. W. Marcellin, and M. Wang. "Trellis-coded vector quantization". *IEEE Transactions on Information Theory*, 37(6):1551–1566, Nov. 1991.

[73] G. Ungerboeck. "Channel coding with multilevel/phase signals". *IEEE Transactions on Information Theory*, IT-28:55–67, Jan. 1982.

[74] C. Bei and R. M. Gray. "Simulations of vector trellis encoding systems". *IEEE Transactions on Communications*, COM-34(3):214–218, March 1986.

[75] M. O. Dunham and R. M. Gray. "An algorithm for the design of labeled-transition finite-state vector quantizers". *IEEE Transactions on Communications*, COM-33(1):83–89, January 1985.

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

[77] C. S. Kim, J. Bruder, M. Smith, and R. Mersereau. "Subband coding of color images using finite state vector quantization". In *Proceedings ICASSP-88 (IEEE International Conference on Acoustics, Speech and Signal Processing)*, pages 753–756, 1988.

[78] T. Kim. "New finite state vector quantizers for images". In *Proceedings ICASSP-88 (IEEE International Conference on Acoustics, Speech and Signal Processing)*, pages 1180–1183, 1988.

[79] C. Manikopoulos and H. Sun. "Finite state vector quantization with an interleaved image source designed for reconstruction of lost packets". In *Proceedings ICASSP-89 (IEEE International Conference on Acoustics, Speech and Signal Processing)*, pages 1894–1897, 1989.

[80] F. Jelinek and J. B. Anderson. "Instrumentable tree encoding of information sources". *IEEE Transactions on Information Theory*, IT-17:118–119, Jan. 1971.

[81] W. A. Finamore and W. A. Pearlman. "Optimal encoding of discrete-time continuoius-amplitude memoryless sources with finite output alphabets". *IEEE Transactions on Information Theory*, IT-26:144–155, Mar. 1980.

[82] J. L. Massey. "Coding and modulation in Digital communications". In *Proceedings of the 1980 Zurich Seminar on Digital Communications*, pages E.2.1 – E.2.3. IEEE Cat.

[83] J. L. Massey. "The how and why of channel coding". In *Proceedings of the 1984 Zurich Seminar on Digital Communications*, pages 67–73. IEEE Cat., No. 84, 1984.

[84] J. Hagenauer. "Soft-in/Soft-out:The benefits of using soft decisions in all stages of digital receivers". In *Proc. of the 3rd Int. Workshop on DSP Technique Applied to Space Communications*, ESTEC Noodrdwijk, The Netherlands, Sept. 1992.

[85] Xiao an Wang and Stephen B. Wicker. "A soft-ouput decoding algorithm for concatenated systems". *IEEE Transactions on Information Theory*, 42(2):543–553, Mar. 1996.

[86] M. A. Herro, D. J. Costello , Jr., and L. Hu. "Capacity and cutoff rate calculations for a concatenated coding system". *IEEE Transactions on Information Theory*, IT-34:212–222, Mar. 1988.

[87] A. Viterbi. "Wireless digital communication: A view based on three lessons learned ". *IEEE Communications Magazine*, 29(9):33–36, Sep. 1991.

[88] L. N. Lee. "Real-time minimum-bit-error probability decoding of convolutional codes". *IEEE Transactions on Communications*, COM-22:146–151, Feb. 1974.

[89] C. R. P. Hartmann and L. D. Rudolph. "An optimum symbol-by-symbol decoding rule for linear codes". *IEEE Transactions on Information Theory*, IT-22:514–517, Sept. 1976.

[90] J. Anderson. "The turbo coding scheme". In *Proceedings ISIT-94 (IEEE International Symposium on Information Theory)*, Trondheim, Norway, June 1994.

[91] J. Hagenauer, P. Robertson, and L. Papke. "Iterative (turbo) decoding of systematic convolutional codes with the MAP and SOVA algorithms". In *Proceedings of ITG Conference*, pages 1–9, Frankfurt, Germany, Oct. 1994.

[92] S. S. Pietrobon and S. A. Barbulescu. "A simplification of the modified Bahl decoding algorithm for systematic convolutional codes". In *Proceedings of ISITA-1994*, pages 1073–1077, Sudney, NSW, Nov. 1994.

[93] J. Hagenauer and P. Hoeher. "A Viterbi algorithm with soft-decision outputs and its applications". In *Proceedings GLOBECOM-88*, pages 47.1.1–47.1.7, Dallas, TX-USA, Nov. 1988.

[94] L. E. Baum and J. A. Egon. "An inequality with applications to statistical estimation for probabilistic functions of a Markov process and to a model for ecology". *Bull. Amer. Meteorol. Soc.*, 73:360–363, 1967.

[95] L. E. Baum and G. R. Sell. "Growth functions for transformations on manifolds". *Pac. J. Math.*, 27(2):211–227, 1968.

[96] L. R. Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition". *Proceedings of the IEEE*, 77(2):257–286, Feb. 1989.

[97] D. Miller, K. Rose, and P. Chou. "Deterministic annealing for trellis quantizater and HMM design using Baum-Welch re-estimation". In *ICASSP'94*, pages V261–V264, Adelaide, Australia, April 1994.

[98] K. Yoon, Y. Ham, and R. Park. "Hybrid approaches to frontal view face recognition using the hidden Markov model and neural network". *Pattern Recognition*, 31(3):283–293, March 1998.

[99] M. Park and D. Miller. " Low-delay optimal MAP state estimation in HMM's with application to symbol decoding". *IEEE Signal Processing Letters*, 4(10):289–292, Oct. 1997.

[100] W. Turin. " Unidirectional and parallel Baum-Welch algorithms". *IEEE Transactions on Speech & Audio Processing*, 6(6):516–523, Nov. 1998.

[101] V. Krishnamurthy and R. J. Elliott. " Filtered EM algorithm for joint hidden Markov model and sinusoidal parameter estimation". *IEEE Transactions on Signal Processing*, 43(1):353–358, Jan. 1995.

[102] J. S. Bridle and L. Dodd. " An Alphabet approach to optimising input transformations for continuous speech recognition". In *Proceedings ICASSP-91 (IEEE International Conference on Acoustics, Speech and Signal Processing)*, volume 1, pages 277–280, 1991.

[103] Tariq Haddad and Abbas Yongaçoğlu. "Symbol-MAP-based trellis vector quantization". In *Proceedings of GLOBECOM'99*, Rio de Janeiro, Brasil, December 5-9 1999.

[104] Tariq Haddad and Abbas Yongaçoğlu. "Fuzzy soft trellis vector quantization". *Submitted for publication to IEEE Trans. on Communications*, Oct. 1999.

[105] P. Robertson, E. Villebrun, and P. Hoeher. "A comparison of optimal and suboptimal MAP decoding algorithms operating in the Log domain". In *Proceedings ICC-95 (IEEE International Conference on Communications)*, pages 1009–1013, Seattle, Washington, USA, June 18-22 1995.

[106] A. J. Viterbi. "An intuitive justification and a simplified implementation of the MAP decoder for convolutional codes". *IEEE Journal on Selected Areas in Communications*, 16(2):260–264, Feb. 1998.

[107] E. Kuruoglu and E. Ayanoglu. "Design of trellis waveform coders with near-optimum structure". *Electronics Letters*, 28(18):1727–1729, Aug. 1992.

[108] C. Nassar and M. Soleymani. " Globally optimal trellis quantizers". In *Proceedings ICASSP-91 (IEEE International Conference on Acoustics, Speech and Signal Processing)*, volume 1, pages 41–44, Toronto, Ont, Can., 1991.

[109] W. H. Equitz. "A new vector quantization clustering algorithm". *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(10):1568–1575, Oct. 1989.

[110] C. Torres and E. Arias. " Stochastic vector quantization of images". *Signal Processing*, 62(3):291–301, Nov. 1997.

[111] H. Monawer. " Image vector quantization using a modified LBG algorithm with approximated centroids". *Electronics Letters*, 31(3):174–175, Feb. 1995.

[112] V. Delport and M. Koschorreck. "Genetic algorithm for codebook design in vector quantisation". *Electronics Letters*, 31(2):84–85, Jan. 1995.

[113] B. Fritzke. " LBG-U method for vector quantization - an improvement over LBG inspired from neural networks". *Neural Processing Letters*, 5(1):35–45, Feb. 1997.

[114] B. Ramamurthi and A. Gersho. "Classified vector quantization of images". *IEEE Transactions on Communications*, COM-34(11):1105–1115, Nov. 1986.

[115] P. Westerink, D. Boekee, J. Biemond, and J. Woods. "Subband coding of image using vector quantization". *IEEE Transactions on Communications*, 36(6):713–719, June 1988.

[116] N. M. Nasrabadi and R. A. King. "Image coding using vector quantization: A review ". *IEEE Transactions on Communications*, 36(8):957–971, August 1988.

[117] N. M. Nasrabadi and Y. Feng. "Image compression using address-vector quantization". *IEEE Transactions on Communications*, 38(12):2166–2173, Dec. 1990.

[118] A. Kurtenback and P. Wintz. "Quantizing for noisy channels". *IEEE Transactions on Communications*, COM-17:291–302, Apr.

[119] N. Rydbeck and C. W. Sundberg. "Analysis of digital errors in nonlinear PCM systems". *IEEE Transactions on Communications*, COM-24:59–65, Jan. 1976.

[120] J. H. Chen, G. Davidson, A. Gersho, and K. Zeger. "Speech coding for the mobile satellite experiment ". In *Proceedings ICC-87 (IEEE International Conference on Communications)*, pages 756–763, Seattle, WA, June 1987.

[121] K. A. Zeger and A. Gersho. "Zero redundancy channel coding in vector quantization". *Electronics Letters*, 23:654–655, June 1987.

[122] H. Kumazawa, M. Kasahara, and T. Namekawa. "A construction of vector quantizers for noisy channels". *Electronics and Engineering in Japan*, 67-B(4).

[123] S. Gadkari and K. Rose. "Robust Vector Quantizer Design by Noisy Channel Relaxation". *IEEE Transactions on Communications*, 47(8):1113–1122, Aug. 1999.

[124] John G. Proakis. *Digital Communications.* McGraw-Hill, Inc., New York, 1995.

[125] R. J. McEliece. *The theory of information and coding.* "Addison-Wesley", 1977.

[126] D. Miller and K. Rose. "Combined source-channel Vector quantization using deterministic annealing". *IEEE Transactions on Communications*, 42(2/3/4):347–356, Feb./Mar./Apr. 1994.

[127] Min Wang and Thomas Fischer. "Trellis-coded quantization designed for noisy channels". *IEEE Transactions on Information Theory*, 40(6):1792–1802, Nov. 1994.

[128] Tariq Haddad and Abbas Yongaçoğlu. "Performance of joint source/channel decoding with dual-k codes". In *TRIO/ITRC Research Retreat*, Kingston, Ontario, "May 6-8" 1997.

[129] Tariq Haddad and Abbas Yongaçoğlu. "Joint source/channel symbol-MAP detection of trellis vector quantized sources". In *Proc. 19th Biennial Symposium on Commun.*, pages 169–173, Kingston, Ontario, "May 31 - June 3" 1998.

[130] Tariq Haddad and Abbas Yongaçoğlu. "Joint source/channel soft Viterbi decoding". In *Proceedings of the Fourth IEEE Symposium on Computers & Communications*, Red Sea, Egypt, 6-8, July 1999.

[131] A. Netravali and J. Limb. "Picture coding: a review". *Proc. of the IEEE*, 68(3):366–407, Mar. 1980.

[132] R. Link and S. Kallel. "Markove model aided decoding for DPCM image transmission". In *Proc. 19th Biennial Symposium on Commun.*, pages 173–178, Kingston, Ontario, "May 31 - June 3" 1998.

[133] A. Aksu and M. Salehi. "Multistage trellis coded quantization (ms-tcq) design and performance". *IEEE Transactions on Communications*, 144:61–64, April 1997.

[134] A. Aksu and M. Salehi. "Design, performance and complexity analysis of residual trellis-coded vector quantization". *IEEE Transactions on Communications*, 46:1020–1026, Aug. 1998.

[135] H. Arda Aksu and M. Salehi. "Joint optimization of TCQ-TCM systems". *IEEE Transactions on Communications*, 44(5):529–533, May 1996.

[136] S. Benedetto and G. Montorsi. "Iterative decoding of serially concatenated convolutional codes". *Electronics Letters*, 32(13):1186–1187, June 1996.

[137] G. Battail. "Coding for the Gaussian channel: the promise of weighted output decoding". *Int. J. Sat. Commun.*, 7:183–192, 1989.

[138] J. Lodge, P. Hoeher, and J. Hagenauer. "The decoding of multidimensional codes using seperable MAP 'filters' ". In *Proc. 16th Bienial Symp. Commun.*, pages 343–346, Queens University, Kingston, Canada, May 1992.

[139] H. Stark and J. Woods. *Probability, random processes and estimation theory for engineers.* Prentice-Hall, New York, 1994.

[140] J. Makhoul. "Linear prediction - A tutorial review". *Proceedings of the IEEE*, 63:561–580, April 1975.

[141] B. Bunin. "Rate-distortion functions for Gaussian Markov processes". *Bell Syst. Tech J.*, pages 3059–3075, Nov.

[142] S. Ghoniemy and S. F.Bahgat. " Rate distortion function for nth-order Gaussian Markov process". *Circuits Systems & Signal Processing*, 12(4):567–578, 1993.

[143] L. D. Davisson. "Rate-Distortion Theory and Application". *Proc. IEEE*, 60:800–808, Jul. 1972.

[144] T. M. Cover and J. A. Thomas. *Elements of information theory.* John Wiley & Sons, New York, 1991.

[145] A. Spanias. "Speech coding: A tutorial review". *Proceedings of the IEEE*, 82(10):1541–1582, Oct. 1994.

[146] A. Ortega and K. Ramchandran. "Rate-distortion methods for image and video compression". *IEEE Signal processing Magazine*, pages 23–50, Nov. 1998.

[147] I. D. Marsland and P. T. Mathiopoulos. "Multiple differential detection of parallel concatenated convolutional (turbo) codes in correlated fast rayleigh fading ". *IEEE Journal on Selected Areas in Communications*, 16(2):265–274, Feb. 1998.

[148] D. Makrakis, P. T. Mathiopoulos, and D. P. Bouras. "Optimal decoding of coded PSK and QAM signals in correlated fast fading channels and AWGN: A combined envelop, multiple defferential and coherent detection approach ". *IEEE Transactions on Communications*, 42(1):63–75, Jan. 1994.

[149] G. M. Vitetta and D. P. Taylor. "Maximum likelihood sequence estimation of uncoded and coded PSK signals transmitted over rayleigh flat-fading channels ". In *Proceedings ICC-94 (IEEE International Conference on Communications)*, pages 1—7, 1994.