Trellis-Coded Quantization with Unequal Distortion

KWONG Cheuk Fai

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of
Master of Philosophy of
Information Engineering

The Chinese University of Hong Kong
July 2001

Trellis-Coded Quantization with Unequal Distortion

KWONG Cheuk Fai

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of
Master of Philosophy

Division of Information Engineering
The Chinese University of Hong Kong
May 2001

Acknowledgements

I would like to express my greatest gratitude to my supervisor Prof. Ricky Keang-Po Ho for his sustained support, beneficial guidance and genial encouragement. He always gives me prompt inspiration with his profound knowledge. His ennobled attitude to research and amiable personality make my study a delightful experience.

And, I would like to thank my co-supervisor, Prof. Raymond Wai-Ho Yeung for his kindly help and support.

Thanks for my colleagues who make my life here unforgettable. Special thanks to Man-Shing Ho, Anthony Wong and Hai-Tao Yu who always give me their greatest help.

Finally, I have to give thanks to my family and Maisan for their love end endless support.

Abstract

Both trellis-coded quantizer (TCQ) and trellis-coded vector quantizer (TCVQ) with unequally fine and coarse distortion are designed for Gaussian sources. The fine and coarse quantizers use the same trellis structure but with different codebooks. The distortion of the fine codebook is smaller than the minimum achievable distortion of the same encoding rate. However, as expected, the average of the fine and coarse distortion is slightly worse than the minimum achievable distortion of the same encoding rate.

Encoding with fine and coarse codebooks alternatively through out the same trellis, the performance of unequal error protection is investigated on scalar TCQ, 2-dimensional TCVQ and 3-dimensional TCVQ, at different rates. Unequal distortion TCQ and TCVQ scheme on both memoryless and Markov Gaussian sources are studied.

摘 要

本論文針對高斯信源設計了具有不同的精確和粗略失真度的格架編碼量化器（TCQ）及格架編碼矢量量化器（TCVQ），它們採用了相同的格架編碼結構（Trellis），但使用了不同的碼本（Codebook）。精確碼本的失真度要比在相同編碼速率下的最小失真度還要小。但是，使用精確和粗略兩種失真度碼本的平均效果只比相同情況下的最小失真度差一些。

通過在相同格架編碼方式下選用這兩種碼本，本文在不同的編碼速率下對標量 TCQ，二維 TCVQ 和三維 TCVQ 的不均勻差錯編碼的保護性能進行了分析，並研究了在無記憶高斯信源和馬爾科夫高斯信源情況下 TCQ 與 TCVQ 兩者的不均勻差錯編碼的特性。

Contents

Chapter 1

Introduction

Using the concept of information refinement [1], many types of signal compression can be provided with unequal distortion. Well-known techniques [2]-[3] such as multi-resolution signal compression or multi-stage vector quantizers (VQ) are used for the purpose of unequal distortion. For example, a multi-stage vector quantizer consists of successively approximating the input vector in several cascaded VQ stages, where the input vector from each stage is the quantization error from the preceding stage. In this way, multi-stage VQ serves as a sub-optimal VQ scheme with reduced complexity and storage. One immediate application of this rate scalability is in progressive transmission.

In this thesis, another method for unequal distortion is studied using the same quantizer. The main idea is to use two different codebooks with unequal resolution on the same trellis for trellis-coded quantization (TCQ) [4].

1.1 Quantization

Without any distortion, an infinite number of bits per sample are required to represent an analog source without any distortion. It is impossible for a practical digital communication system to transmit infinite number of bits. Therefore, analog source is mapped into a finite set of reproductive symbols to restrict the bit rate to a practical level. However, this process will introduce distortion to the analog signal. The process of representing the analog source using finite number of symbols is called quantization [5] and the distortion is called quantization error. The quantization function Q is defined by

$$Q(x) = c_k, \quad k = 1,...M, \tag{1.1}$$

where c_k is the codebook, and M is the number of symbols in the codebook. The function of $Q(x)$ is a non-linear and non-invertible (many-to-one mapping) function. Quantization error D is defined as

$$D = E[(x - Q(x))^2], \tag{1.2}$$

which is the squared distance between the source samples and the reproductive symbols.

An optimal quantizer provides a mapping function such that D is minimized. Rate distortion theory suggests that there is a tradeoff between bit rate and distortion. Given a coding rate R, a distortion limit D can be found for different source [6].

Quantization can operate on a group (or vector) of analog sources or a single analog source, namely, vector and scalar quantization, respectively. The quantization level $Q(x)$ is a vector or a scalar for vector and scalar quantization, respectively. It is obvious that scalar quantization is a special case for vector quantization. In scalar quantization [5][7], each single source is quantized into a number of levels and these levels are then encoded into a binary sequence. In vector quantization [2][8], a number of source samples are grouped as a vector and then quantized into a number of vectors (a finite set of vectors), called code vector. Information theory suggests that quantization can always achieve better performance by coding vector instead of scalar analog source. However, the complexity of vector quantization is always higher than that of scalar quantization. Scalar quantization is still widely used in many applications such as audio, video and waveform coding [9].

1.2 Trellis-Coded Quantization

Trellis-Coded Quantization (TCQ) [4] is an efficient method for encoding analog sources. The quantization distortion performance of TCQ is excellent with modest

complexity. TCQ is a form of trellis coding that was motivated by Ungerboeck's formulation [5] of trellis-coded modulation (TCM) using a structured codebook with expanded set of quantization levels. While performance of TCQ is in many cases close to the theoretical rate-distortion bound, an improvement is always possible by generalizing its structure to the vector case. Trellis-coded vector quantization (TCVQ) is the multi-dimensional extension of TCQ.

1.3 Thesis Organization

In this thesis, we studied TCQ with unequal distortion. The design is based on unequal resolution codebooks for TCQ to provide unequal distortion. The performance measurement is based on the mean square error distortion. Simulations are performed on memoryless and Markov Gaussian source. Both scalar TCQ and TCVQ cases will be studied. The thesis organization is described as follows.

Chapter 2 gives a brief summary about trellis coding starting from convolutional codes to TCM. The basic of trellis diagram and Ungerboeck codes will be introduced.

The background of TCQ and TCVQ will be covered in Chapter 3. Details of set partitioning and codebook optimization will be discussed. These techniques will be used in later chapters.

An unequal error protection TCQ coder is designed in Chapter 4. The design procedures and issues are discussed.

Performance of unequal distortion TCQ on memoryless Gaussian source is measured and presented in Chapter 4. Set partitioning of a codebook for memoryless Gaussian source is demonstrated.

The simulation of unequal distortion TCQ is performed on Markov Gaussian source in Chapter 6. Performance of unequal distortion TCQ coder on Gaussian sources with different degree of correlation is discussed.

Lastly, a conclusion comes in Chapter 7 as a summary and discussion on applications of the unequal distortion TCQ coder.

Chapter 2
Trellis-Coded Modulation

Trellis-coded modulation (TCM) is a combined coding and modulation scheme for improving the reliability of a digital transmission system without increasing the transmitted power or required bandwidth [10].

In all communication systems, both power and bandwidth are expensive resources. For a power-limited system, the desire system performance should be achieved with the smallest possible power. Error-correcting codes increase the power efficiency by adding extra bits to the transmitted symbol sequence, and this procedure requires the modulation system to be operated at a higher data rate and thus at a larger bandwidth. For the bandwidth-limited system, frequency utilization can be improved by choosing higher order modulation schemes (e.g. 8-PSK instead of 4-PSK), but a larger signal power would be needed to maintain the same signal separation and hence the same error probability. TCM is the solution that combines the choice of a higher order modulation scheme with that of a convolution code, instead of performing demodulation and decoding in two separate steps in the receiver, the two operations are combined into one.

When the receiver combines demodulation and decoding in a single step, the performance of the transmission system do not depends on the free Hamming

distance of the convolutional code, but rather, depends on the free Euclidean distance between transmitted signal sequences of the additive white Gaussian noise channel. The optimization of the TCM design is based on Euclidean distances rather than on Hamming distances. The choice of the code and signal constellation cannot be performed separately. Encoding and decoding of TCM will be discussed in later sections and it is started by an introduction of convolutional code.

2.1 Convolutional Codes

There are two basic kinds of error-correcting codes, namely block codes and sliding window codes. A convolutional code, denoted by (k, n, λ) is an error-correcting code with sliding window.

Convolutional codes differ from block codes in that the encoder contains memory and the n-bit encoder outputs at any given time unit depend not only on the k-bit inputs at that time unit but also on λ previous input blocks [10][12]. A (k, n, λ) convolutional code can be implemented with a k-input, n-output linear sequential circuit with input memory size λ. Typically, n and k are small integers with $k < n$, but the memory order λ can be large to achieve low error probabilities. In the important special case when $k = 1$, the information sequence is not divided into blocks and can be processed continuously.

The stream of information bits is partitioned into blocks of k bits ($\boldsymbol{u}$) each. Each block of k information bits is encoded into n coded bits ($\boldsymbol{v}$). We call $\boldsymbol{u}$ and $\boldsymbol{v}$ as input and output symbol respectively (Figure 2.1). The rate of the convolutional code is k/n bits per sample where $(n-k)$ bits are added as redundancy. Moreover, the encoded symbol $\boldsymbol{v}$ not only depends on the current block but also the λ pervious blocks. In other words, the convolutional coder has memory λk bits.

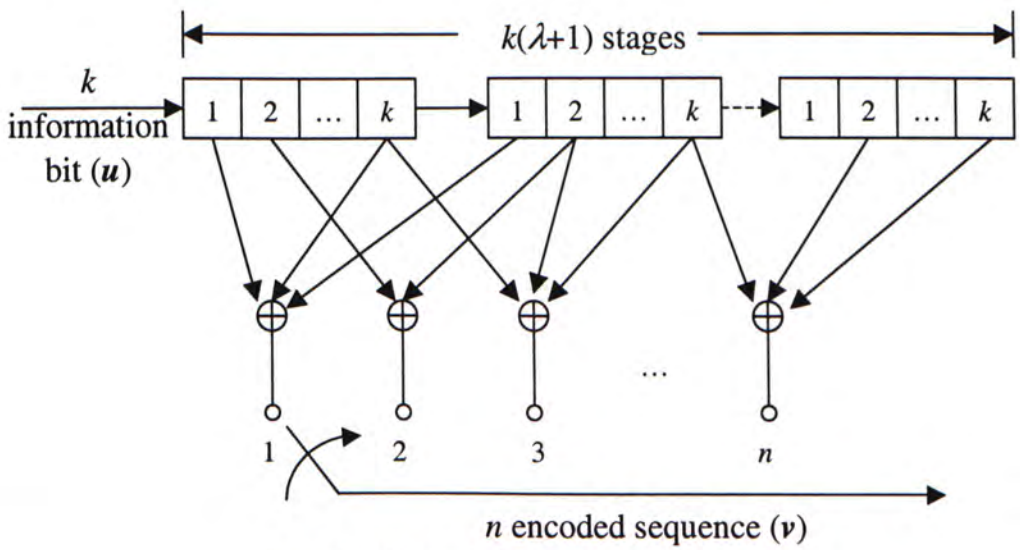


Figure 2.1 A (k, n, λ) Convolution Encoder.

2.1.1 Generator Polynomials and Generator Matrix

If

$$u(D) = \sum_{i=0}^{\infty} u_i D^i \quad \text{or } \boldsymbol{u} = (u_0, u_1, u_2, \ldots) \tag{2.1}$$

denotes input sequence to the encoder and

$$v(D) = \sum_{i=0}^{\infty} v_i D^i \text{ or } \boldsymbol{v} = (v_0, v_1, v_2, \ldots) \tag{2.2}$$

denotes the output sequence where the indeterminate D is unit delay operator. Then the input-output relationships are expressed

$$v^{(i)}(D) = u(D) \times G_i(D), \text{ where } \; G_i(D) = \sum_{j=0}^{\lambda} g_{i,j} D^j \tag{2.3}$$

and they are called the *generator polynomials* of convolutional code. For (n, k, λ) code, there will have n generator polynomials.

The convolutional code is linear such that the coding equation can be described by the matrix multiplication. The generator polynomial can be organized into a semi-infinite matrix $\boldsymbol{G}$, which is called *generator matrix*. The input-output relationships are defined as

$$\boldsymbol{v} = \boldsymbol{u}\boldsymbol{G} \tag{2.4}$$

where

$$G = \begin{bmatrix} G_0 & G_1 & G_2 & \cdots & G_\lambda & & \\ & G_0 & G_1 & G_2 & \cdots & G_\lambda & \\ & & G_0 & G_1 & G_2 & \cdots & G_\lambda & \\ & & & \ddots & & \cdots & & \ddots \end{bmatrix} \tag{2.5}$$

and each G_l is a k by n matrix specified below:

$$G_l = \begin{bmatrix} g_{1,l}^{(1)} & g_{1,l}^{(2)} & \cdots & g_{1,l}^{(n)} \\ g_{2,l}^{(1)} & g_{2,l}^{(2)} & \cdots & g_{2,l}^{(n)} \\ \vdots & \vdots & \ddots & \vdots \\ g_{k,l}^{(1)} & g_{k,l}^{(2)} & \cdots & g_{k,l}^{(n)} \end{bmatrix} \tag{2.6}$$

where $g_i^{(j)} = (g_{i,0}^{(j)}, g_{i,1}^{(j)}, g_{i,3}^{(j)}, \cdots g_{i,\lambda}^{(j)}), 1 \le i \le k, 1 \le j \le n$, are the *generator sequences*.

The generator matrix is used to specify a convolutional code.

2.1.2 Circuit Diagram

The convolutional code encoder can be implemented by a linear sequential circuit. The circuits consist of a shift register with $k\lambda$ stages. At each instant of time, k information bits ($\boldsymbol{u}$) enter the shift register and the contents of the last k states of the shift register dropped. The n-bit output sequence ($\boldsymbol{v}$) is generated by the k-bit input together with the memory of λ stages, stored in the delay. The encoder depends on not only the k bits of input but also the $k\lambda$ contents from the memory. Figure 2.2

shows a circuit diagram of a convolutional code encoder with generator polynomials $g_1 = 1 + D^2$ and $g_2 = 1 + D + D^2$.

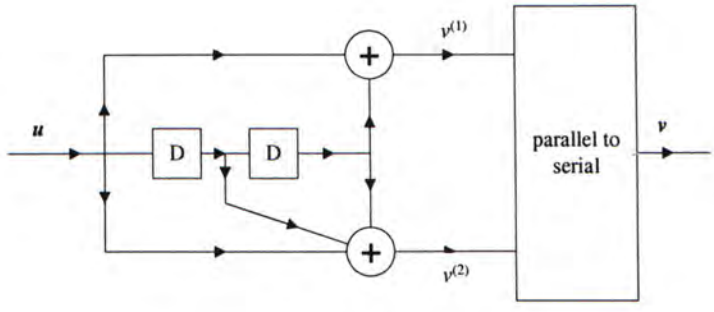


Figure 2.2 A circuit diagram of (1,2,2) convolutional encoder.

2.1.3 State Transition Diagram

Convolutional code can be represented as a finite-state machine. The total number of states is equal to $2^{k\lambda}$ where $k\lambda$ is number of stages in the coder. Given the input symbol $\boldsymbol{u}$ and the current state s, the output symbol $\boldsymbol{v}$ is calculated and the state is changed from s to s'. Figure 2.3 shows the state transition for the (1, 2, 2) convolutional code in Figure 2.2. The state transition edges are labeled by *a/b* where *a* and *b* represent the *k*-bit input and the *n*-bit output, respectively.

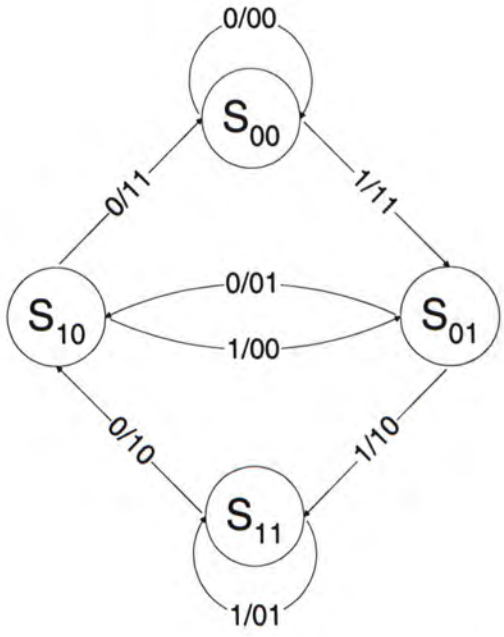


Figure 2.3 State transition diagram of the (1, 2, 2) convolutional code in Fig. 2.2.

2.1.4 Trellis Diagram

A *trellis diagram* is used to show the various states and coded sequence of a convolutional code as time evolves. It shows the possible transitions and the coding path as well. Each block of k inputs causes a transition to a new state. Hence, there are 2^k branches leaving each state, one corresponding to each different input block. For a $(1, n, \lambda)$ convolutional code, there are only two branches leaving each state. Each branch is labeled with the k inputs causing the transition and n corresponding outputs. Figure 2.4 depicts a trellis diagram of Figure 2.2 with a coding path. We will use trellis diagram in later chapters for analysis in encoder and decoder.

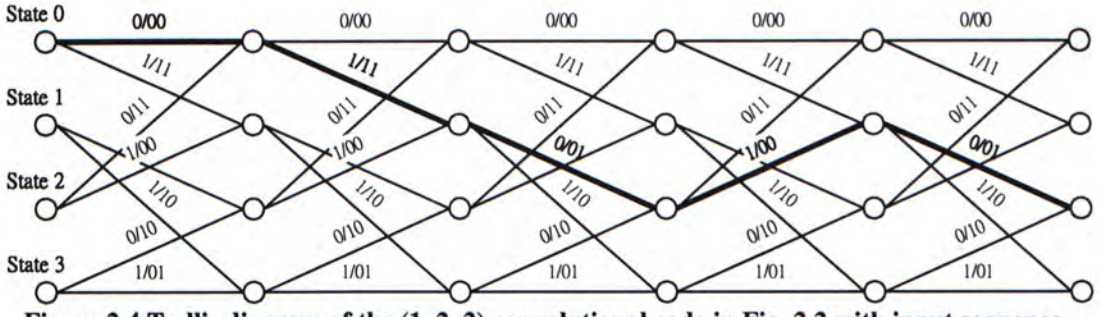


Figure 2.4 Trellis diagram of the (1, 2, 2) convolutional code in Fig. 2.2 with input sequence $u = [0,1,0,1,0\ldots]$ and output sequence $v = [00,10,01,00,01\ldots]$.

2.2 Trellis-Coded Modulation

In a communication system, messages are represented by vectors in an N-dimensional Euclidean space $\mathbf{R}^N$, called the *signal space*. When the vector x is transmitted, the received signal is represented by the vector

$$y = x + n \tag{2.7}$$

where n is a noise vector whose components are independent Gaussian random variables with zero mean and the same variance $N_0/2$. The vector x is represented by a set Ω' consisting of M' signal vectors, the *signal constellation*. The average square signal *energy* is

$$E' = \frac{1}{M'} \sum_{x \in \Omega'} \|x\|^2 \tag{2.8}$$

Consider now the transmission of a sequence $\{x_i\}_{i=0}^{K-1}$ of signals of length K, where

the subscript i denotes discrete time. For the received sequence $y_0,\ldots y_{K-1}$, to minimize the average error probability, the receiver decodes and interprets the output sequence $\hat{x}_0,\ldots,\hat{x}_{K-1}$ (the decoded signals are also represented by the signals from the set Ω') if

$$d^2 = \sum_{i=0}^{K-1} \left\| y_i - \hat{x}_i \right\|^2 \tag{2.9}$$

is minimized, or in other words, if the sequence $\hat{x}_0,\ldots,\hat{x}_{K-1}$ is closer to the received sequence than to any other allowable signal vector sequence. The resulting sequence error probability, as well as the symbol error probability, is upper bounded, at least for high signal-to-noise ratios, by a decreasing function of the ratio $d_{\min}^2 / N_0$, where $d_{\min}^2$ is the minimum squared Euclidean distance between two allowable signal vector sequences.

2.2.1 Uncoded Transmission versus TCM

A simple case occurs when the signals form an *independent* sequence, i.e. an uncoded sequence. The allowable signal sequences are all the elements of Ω'^K (K is the length of the signal), and hence d^2 is minimized by minimizing separately the individual terms $\left\| y - \hat{x} \right\|^2$ for $\hat{x} \in \Omega'$. The performance of this symbol-by-symbol receiver will then depend on the minimum distance

$$d_{\min}^2 = \min_{x' \neq x''} \left\| x' \neq x'' \right\|^2 \tag{2.10}$$

as x', x'' run through Ω'. In fact, the symbol error probability is upper bounded by

$$P(e) \leq \frac{M'-1}{2} \operatorname{erfc}\left(\frac{d_{\min}}{2\sqrt{N_0}}\right), \tag{2.11}$$

where N_0 is the energy of the noise.

The performance [9] can be improved significantly with the coded sequence. The transmitted sequences can be restricted to a subset of Ω'^K. The transmission rate will also be reduced using just a subset of the space. To avoid the reduction of transmission rate, the size of Ω' can be increased. For example, if the signal space is changed from Ω' to $\Omega \in \Omega'$ and M' to $M > M'$, and M'^K sequences are selected as a subset of Ω^K, with the minimum distance between the sequences can be increased.

A minimum distance d_{free} between two possible sequences can be greater than the minimum distance $d_{\min}$ between signals in Ω', the original signal space. To avoid a reduction of the value of the transmission rate, the constellation is expanded from Ω' to Ω. This may entail an increase in the average energy expenditure from E' to E. The *asymptotic coding gain* of a TCM scheme can be defined as

$$\gamma = \frac{d_{\text{free}}^2 / E}{d_{\min}^2 / E'} \tag{2.12}$$

where E' and E are the average energy spent to transmit with uncoded and coded transmission, respectively.

Assumed that the signal x_n transmitted at discrete time n depends not only on the source symbol a_n transmitted at the same time instant (as it would be with memoryless modulation), but also on a finite number of previous source symbols:

$$x_n = f(a_n, , a_{n-1}, \ldots, a_{n-L}) \tag{2.13}$$

By defining

$$s_n = (a_{n-1}, \ldots , a_{n-L}) \tag{2.14}$$

as the *state* of the encoder at time n, we can rewrite the formulae in the more compact form

$$x_n = f(a_n, , s_n) \tag{2.15}$$

$$s_{n+1} = g(a_n, s_n) \tag{2.16}$$

Assume that the functions f and g are time invariant, the function $f(\cdot,\cdot)$ depends not only on the corresponding source symbol, but also on current state s_n. In other words, at any time instant the transmitted symbol is chosen from a constellation that is selected by the value of s_n (i.e. the current state). The function $g(\cdot,\cdot)$ describes the memory part of the encoder and shows the evolution of the modulator states, as shown in Figure 2.5.

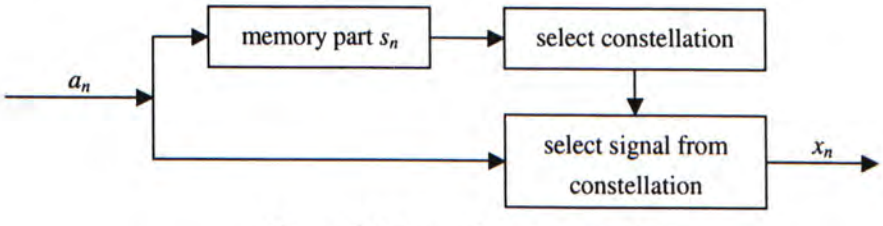


Figure 2.5 General model for TCM.

2.2.2 Trellis Representation

Trellis is the graphical representation of the function f and g, with the parameter s_n, the encoder state at time n as the nodes of the trellis [9][12]. For each source symbol there are branches that stem from state s_n to the next state s_{n+1}. The branch is labeled by the corresponding value f. The trellis structure is determined by the function g, while f describes how channel symbols are associated with each branch along the trellis.

If the source symbols are M'-ary, each node must have M' branches stemming from it (one per each source symbol). In some cases two or more branches connect the same pair of nodes; and it is called *parallel transitions*. For uncoded transmission, the trellis degenerates to a single state and thus *all transitions* are parallel transitions.

Figure 2.6 shows a trellis with four states, and a constellation with four signals. Optimum decoding is the search of the most likely path through the trellis once the received sequence has been observed at the channel output. This search is best performed by Viterbi algorithm.

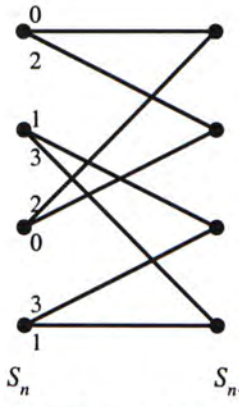


Figure 2.6 A trellis describing a TCM scheme with four states and four channel symbols.

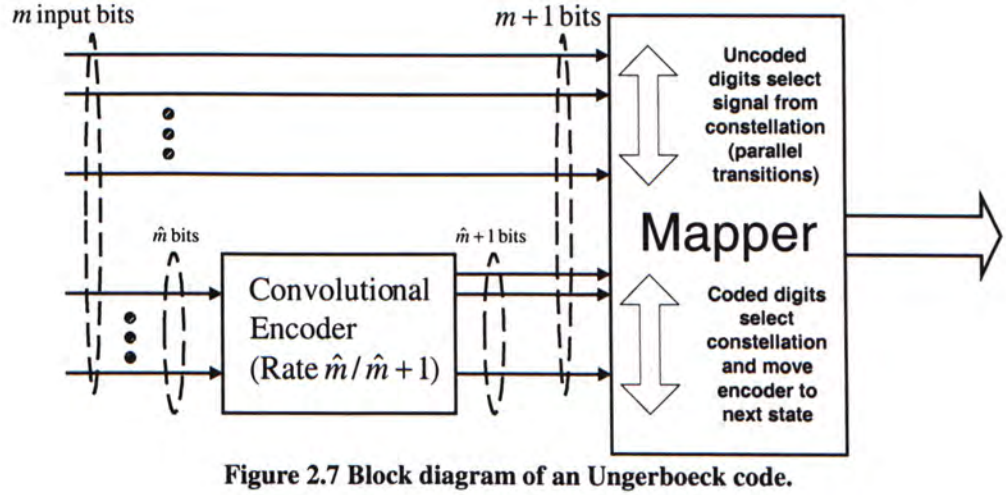


Figure 2.7 Block diagram of an Ungerboeck code.

2.2.3 Ungerboeck Codes

Based on the trellis representation, we shall now describe a special trellis codes called Ungerboeck codes [5][12]. Figure 2.7 shows the Ungerboeck representation of a trellis encoder with a symbol mapper. The mapper is used to map each output

$(m+1)$ bits (symbol v) to a signal constellation. This process is called bits-to-symbol assignments. At each encoding step, the rate $(\hat{m}/\hat{m}+1)$ convolutional encoder receives $\hat{m}$ input bits and generates $(\hat{m}+1)$ coded digits. The coded digits determine the sub-constellation from which the transmitted signal has to be chosen. The other $(m-\hat{m})$ bits left uncoded and represent parallel transitions between two states. Those uncoded bits are used to choose an element from the sub-constellation. The method of grouping the signal constellations into subset is called set partitioning. This method is used to maximize possible Euclidean distance between the signal symbols assigned to same subset and to improve reliability of the trellis codes.

2.2.4 Set Partitioning

Referring to Figure 2.5, a constellation of signals is selected first and a signal is selected as output according to the input. Details shown in Figure 2.5 that $\hat{m}$ bits out of the m bits of input determine the next state as well as the constellation, as a subset of signals constellation. The idea of set partitioning comes from the maximization of the free distance d_{free}, the distance between the signals associated with a pair of paths that originate from an initial split and merge into a single node after L time units (Figure 2.8).

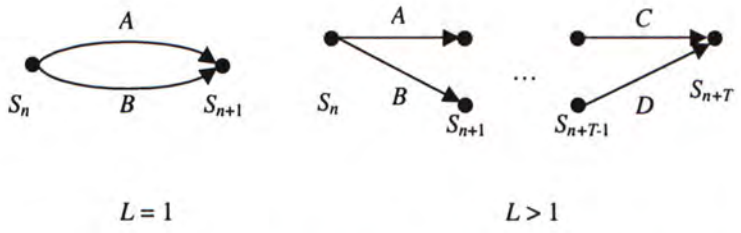


Figure 2.8 Splitting and remerging paths for $L = 1$ (parallel transitions) and $L > 1$.

The free distance d_{free} is determined by parallel transmission for the case of $L = 1$. Consider next $L > 1$, using A, B, C, D to denote subsets of signals associated with each branch, and $d(X, Y)$ denoting the minimum Euclidean distance between one signal in X and one in Y, d_{free} has the expression

$$d_{free}^2 = d^2(A,B) + \cdots + d^2(C,D). \tag{2.17}$$

For an optimized code, subsets assigned to the same originating or to the same terminating state (it is called adjacent transitions or parallel transitions) must have a largest possible distance. Ungerboeck suggested a technique called *set partitioning* to achieve this objective [13].

Set partitioning is used to find a symbol assignment to the branches such that the subsets assigned to the same originating state or to the same terminating state have the largest possible distance. Ungerboeck's set partitioning method is widely used in design of TCM for its ease of use and understanding. The set partition process can be

summarized as the following rules:

1. The Euclidean distance between the signal symbols within the same subset is in maximum possible value. In other words, the distance of signal symbols assigned to the parallel transitions are as large as possible.
2. Signal symbols assigned to the transitions which diverging from or merging into a trellis state will have the next maximum possible Euclidean distance.

Based on the above two rules, the set partitioning principle maximizes the normalized square free distance d^2_{free} which is the minimum distance between two decoding paths. When d^2_{free} is maximum, the reliability of the code is optimal. Ungerboeck demonstrated the set partitioning method for PAM, PSK and QASK signals. Figures 2.9 and 2.10 show examples of partitioning 8-AM and 16-QAM constellations, respectively. This will be used in the following chapters as codebook for scalar TCQ and 2-dimensional TCVQ.

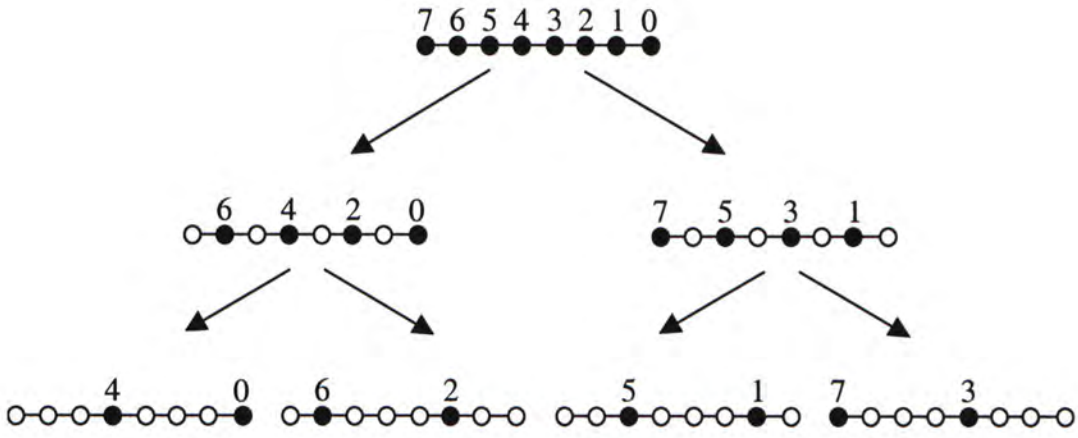


Figure 2.9 Set partitioning of 8-AM channel signals into 4 subsets.

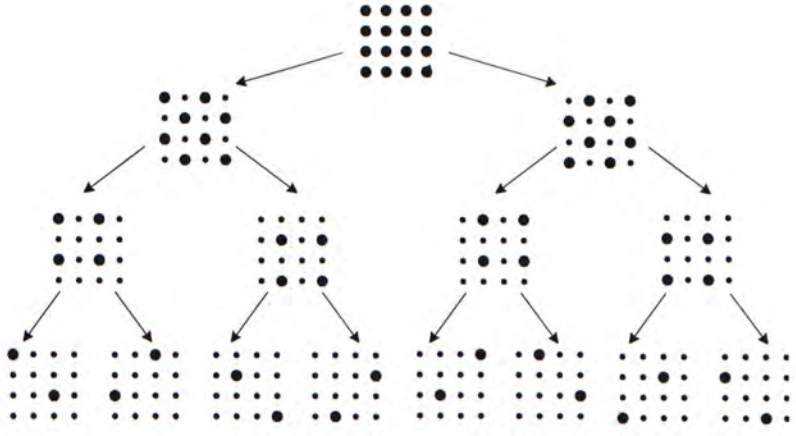


Figure 2.10 Set partitioning of 16-QAM channel signals into 8 subsets.

2.2.5 Decoding for TCM

Encoding and decoding of TCM scheme using a trellis, branches of the trellis are associated with transitions between encoder states and with signals transmitted over

the channel. The task of the TCM decoder is to estimate the path that the encoded signal sequence traverses through the trellis. Each branch of the trellis is associated with a *branch metric* and the path with minimum total metric is chosen as the optimum path for decoding, using the same notation in section 2.2. Consider a sequence of M-ary symbols to be transmitted,

$$\mathbf{x} = \{x_0, x_1, \ldots, x_{K-1}\} \tag{2.18}$$

where each x_i can take on M values. If additive random noise $n(t)$ is added to the transmitted signal, the received signal is in the form of

$$y(t) = x(t) + n(t) \tag{2.19}$$

The task of the demodulator is to process the observed signal $y(t)$ in order to produce an estimate $\hat{\mathbf{x}}$ of the transmitted symbol sequence $\mathbf{x}$,

$$\hat{\mathbf{x}} = \{\hat{x}_0, \hat{x}_1, \ldots, \hat{x}_{K-1}\} \tag{2.20}$$

The *metric* is defined as

$$m[y(t), \mathbf{x}] = \sum_{k=0}^{K-1} \|y_k - x_k\|^2 \tag{2.21}$$

The decoding process is to find the minimum value of the above metric.

For the simplest case that all the information symbols are equally likely (independent

of x_k), all the terms $\sum_{k=0}^{K-1}\|y_k - x_k\|^2$ are independent, the minimum of the sum equals to the sum of the minima

$$\min_{\mathbf{x}} m[y(t),\mathbf{x}] = \sum_{k=0}^{K-1}\min_{x_k}\|y_k - x_k\|^2 \tag{2.22}$$

We can use *symbol-by-symbol demodulation*, in which separate demodulation of the symbol transmitted in each interval is indeed optimum [14]. The demodulation is looking for the value of x_k such that the *Euclidean distance* $\|y_k - x_k\|^2$ between the received signal y_k and the candidate signal x_k is minimum.

If the coded sequence x_k is correlated instead of independent random variables, the symbol-by-symbol demodulation is not the optimal decoder but *Viterbi algorithm.*

The Viterbi algorithm selects the maximum likelihood sequence [15]. The algorithm finds a continuous path which minimums the distortion between the received symbols y_k and the estimated symbols $\hat{m}_k$. TCM receiver can use Viterbi algorithm as the core of coding.

Given a sequence of received symbols $\{y_k\}$, the receiver finds a continuous path with minimum distortion D where

$$D = \frac{1}{K}\sum_{k=0}^{K-1}(y_k - \hat{x}_k)^2 \tag{2.23}$$

is minimum and $\hat{x}_k$ are the decoded sequence of transmitted symbols or equivalently TCM waveforms. The Viterbi algorithm does not minimize the

probability of error of the data bits. However, its performance is very close to that of the optimal symbol-by-symbol decoding algorithms in linear codes [14]. Viterbi algorithm is widely used in decoding convolutional code due to its low cost and low complexity. In geometry representation, Viterbi algorithm is exactly the shortest path algorithm, which finds a path having minimum distortion (cost).

Chapter 3
Trellis-Coded Quantization

3.1 Scalar Trellis-Coded Quantization

Trellis-coded quantization (TCQ) is a form of trellis coding that labels the trellis branches with subsets of reproduction symbols. It was proposed by Marcellin and Fischer in 1990 [4]. TCQ is a "dual" process of TCM. The trellis structure of TCQ is exactly the same as TCM. For a $R/(R+1)$ trellis code, the encoding rate of TCQ is R bits per samples. However, there are 2^{R+1} codewords in the codes and the codebook size is doubled. Each output symbol $\boldsymbol{v}$ of the trellis corresponds to a codeword of the codebook. However, the output of TCQ encoder is not a sequence of output symbols $\boldsymbol{v}$. Instead, the output is the sequence of the symbol $\boldsymbol{u}$, which is R bits per sample, indicating a continuous encoding path of the trellis. TCQ decoder goes through the path again using $\{\boldsymbol{u}\}$ and constructs our targeted codeword sequence.

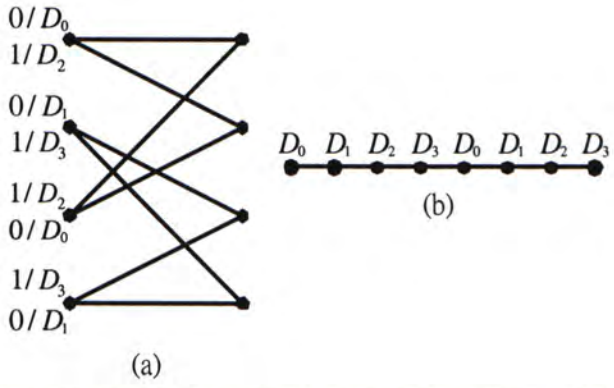
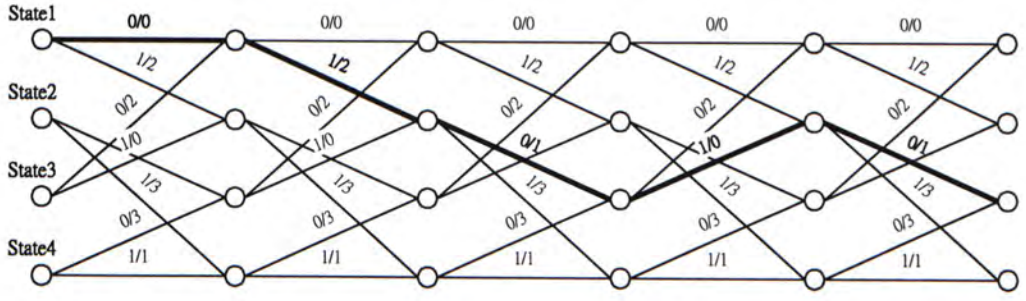


Figure 3.1 (a) Ungerboeck's four-state amplitude modulation trellis. (b) Output points and partition for 2 bits per sample TCQ.

The original TCQ also uses Ungerboeck's method [5] to partition the output points (scalar quantization levels) into subsets $\{D_i\}$. Figure 3.1 shows a 4-state trellis with parallel transitions. The output points are partitioned into four subsets by starting with the leftmost point and proceeding to the right. They are labeled as D_0, D_1, D_2, D_3, D_0, D_1, D_2, D_3, ... until the right most point is reached. Fig. 3.1 is the same as the Ungerboeck's four-state amplitude modulation (4-AM) trellis.

The encoding path is found by Viterbi algorithm, which minimizes the distortion between the output points and the analog input samples. To simplify the idea of TCQ, Figure 3.2 shows an example of TCQ encoding. The input of TCQ is a sequence of real source samples $\boldsymbol{x} = \{x_1, x_2, ..., x_N\}$. Then TCQ finds the sequence of output symbols that minimize distortion $D = \|\boldsymbol{x} - \boldsymbol{v}\|$. The corresponded input symbols $\boldsymbol{u}$ of the trellis is the encoded output symbol of TCQ encoder.



$u = [0,1,0,1,0]$ and $v = [0,2,1,0,1]$

Figure 3.2 An example of TCQ encoding using 4-state trellis with R = 1.

Tables 3.1 and 3.2 show the simulation results for encoding memoryless uniform source and memoryless Gaussian source respectively [4]. They are using Lloyd-Max quantizer output codewords. The performance of Lloyd-Max quantizer is put in here as comparison. An optimal quantizer follows the *centroid condition*: the optimal output level, y_i, for the i^{th} cell of the partition is the centroid of that part of the input pdf . This is observed by Lloyd in 1957 [7] and later by Max [5].

A sequence of 1000,000 source samples is used with block size 1000. Data are measured in terms of signal-to-distortion ratio (SDR) with dB as a unit.

Rate (bits)	Trellis Size (States)							Lloyd-Max Quantizer	Rate Distortion Bound
	4	8	16	32	64	128	256		
1	5.78	5.96	6.06	6.13	6.19	6.29	6.33	6.02	6.79
2	12.47	12.60	12.69	12.76	12.83	12.90	12.93	12.04	13.21
3	18.77	18.90	18.98	19.04	19.10	19.16	19.20	18.06	19.42

Table 3.1 Performance of TCQ for memoryless uniform source (Values are listed as SDR in dB).

Rate (bits)	Trellis Size (States)							Lloyd-Max Quantizer	Rate Distortion Bound
	4	8	16	32	64	128	256		
1	6.22	6.33	6.39	6.44	6.48	6.55	6.58	6.02	6.79
2	12.62	12.73	12.80	12.85	12.91	12.97	13.00	12.04	13.21
3	18.83	18.94	19.01	19.08	19.13	19.18	19.23	18.06	19.42

Table 3.2 Performance of TCQ for memoryless Gaussian source (Values are listed as SDR in dB).

The results show that TCQ can almost provide better performance than the Lloyd-Max scalar quantizer. When the number of state of trellis increases, the performance of TCQ also increases. For examples, TCQ achieves 0.46 dB improvement over Lloyd-Max scalar quantizer at $R = 1$. More states in the trellis give more alternative path for which TCQ encoder can choose. TCQ is close to the theoretical rate-distortion bound. The Viterbi algorithm is used to find the path with minimum distortion.

Here are the steps in designing a TCQ encoder:

1. Selection of a trellis.
2. Selection of codebook.

3. Set partitioning of the codebook.
4. The assignment of subsets to the trellis branches.
5. The optimization of the codebook for a fixed trellis.

Firstly a suitable trellis must be chosen. Referring to Tables 3.1 and 3.2, the SDR increases as the number of states of the trellis increases. Of course, the computation complexity increases as well. A suitable trellis has to be chosen to achieve the targeted SDR performance with acceptable computation complexity.

Secondly, choose a suitable codebook. This codebook serves as an initial codebook to be optimized in step 5. Using the 4-AM trellis as shown in Figure 3.1 as an example, the 8 codewords are distributed evenly on the linear scale. All the symbols on the sequence are quantized into these 8 output symbols only.

Thirdly, the codewords are partitioned into subsets. The objective is to maximize the minimum distance within each subset. Using the 4-AM trellis as example, those 8 codewords are partitioned into four subsets and are denoted as D_0, D_1, D_2, D_3, D_0, D_1, D_2, D_3, in order. The positions denoted as D_0 belong to the same subsets and separated as far as possible in this partition.

At last, assign each subset to trellis branch, and the codewords within each subset represents parallel transitions. Finally, codebook is optimized using generalized Lloyd algorithm. The generalized Lloyd algorithm is discussed in section 3.2.2.

The complexity of TCQ encoder is the same as that of Viterbi algorithm. The encoding requires only 4 multiplications, 4 additions and 4 scalar quantization per source sample, plus 2 additions and 1 comparison per trellis state per source sample. The encoding complexity is roughly independent of the encoding rate. The complexity depends on the number of state per trellis.

TCQ has been applied in image coding [16]-[19], combined with either DCT (discrete cosine transform) or wavelet-based coder and progressive transmission [17]. TCQ often uses with entropy coder to encode memory source. Different types of TCQ such as predictive TCQ [20], embedded TCQ [21] and multistage TCQ [22] had been developed as variation of TCQ.

3.2 Trellis-Coded Vector Quantization

TCQ offers better rate-distortion characteristic than Lloyd-Max scalar quantization by doubling the codebook size at a given code rate. Both TCQ and Lloyd-Max quantizer are all scalar quantization and work well for memoryless source. For correlated sources, vector quantization can provide better rate-distortion performance than that of scalar quantization. In vector quantization, a number of source samples are grouped as a vector and each vector is quantized into a vector codeword. Shannon's rate-distortion theory reveals that coding in vectors instead of scalars can

always achieve better performance, even for memoryless source [6][23].

Combining the idea of vector quantization and trellis-coded quantization, Fisher, Marcellin and Wang [24] proposed trellis-coded vector quantization (TCVQ) in which each codeword is a vector and the codebook size is twice the size of that of a normal vector quantizer. TCVQ can allow non-integer encoding rate. Like TCQ, the main design issues of TCVQ are the design of codebook and the set partitioning of the codebook.

At each encoding step k, the encoder accepts m source samples $\boldsymbol{x}_k = (x_k^{(1)}, \ldots, x_k^{(m)})$, each branches output symbol corresponds to an m dimensional codeword $\boldsymbol{c}_k$. The Viterbi algorithm is employed (as the case of TCQ) to find the most likely path that minimizes the distortion D of

$$D = \frac{m}{N} \sum_{k=1}^{N/m} \left\| \mathbf{x}_k - \mathbf{c}_k \right\|^2 \tag{3.1}$$

Here are the steps in designing a TCVQ encoder:

1. Selection of a trellis.
2. Selection of codebook. The initial codebook is designed by the LBG algorithm, which is purposed by Linde, Buzo and Gary [25] in 1980.
3. Set partitioning of the codebook.
4. The assignment of subsets to the trellis branches.

5. The optimization of the codebook for a fixed trellis.

The steps are the same as that in designing TCQ. The main design issue of TCVQ is the set partitioning of the vector reproduction symbols. The principle of the set partitioning of TCVQ is the same as that of TCQ. The irregular codewords and increased dimensions increase the algorithm complexity. The codebook needs to be optimized as well. Details are discussed as below.

3.2.1 Set Partitioning in TCVQ

The initial codebook of TCVQ is designed by the LBG algorithm [25] to provide optimal codebooks. Afterward, the codewords are partitioned into subsets, as mentioned before. The objective is to maximize the minimum distance within each subset [26]. Given an initial codebook of size 2^{R+1}, the distances between each pair of codewords are measured and sorted in ascending order, and stored in a table. If the ith entry of the distance is denoted as $d_i = |c_i - c_i'|$, where c_i and c_i' are the codewords, and those codewords are partitioned into two subsets, A_0 and A_1, each of size 2^R, the procedures are:

1. c_0 and c_0' are placed into the subsets A_0 and A_1 respectively; remove the entry d_0 from the table.

2. From the table, find the minimum d_i such that c_i or c_i' belongs to $A_0 \cup A_1$.
3. If both c_i and c_i' belongs to $A_0 \cup A_1$, remove the ith entry d_i from the table. Go to step (2).
4. If c_i belongs to A_0 (or A_1), then add c_i' to A_1 (or A_0). If c_i' belongs to A_0 (or A_1), then add c_i to A_1 (or A_0).
5. If the size of either subsets A_0 or A_1 reaches 2^R, then put all the remaining unassigned codewords to the other subset and stop. Otherwise, go to step (2).

By applying the above procedures, the codewords can be partitioned into two subsets. The same procedures can be applied to the two subsets to generate four subsets, and so on. In this thesis, we partition the codebook using the above algorithm.

3.2.2 Codebook Optimization

To minimize quantization distortion, the quantization levels of a codebook can be optimized iteratively using the generalized Lloyd algorithm. Using a training sequence of analog samples, the algorithm is operated as following:

1. An initial codebook is generated by LBG algorithm.

2. Encode the analog training sequence into a sequence of codewords using Viterbi algorithm.
3. Update each codeword with the average of training samples that map to that quantization level, i.e., the centroid of the subset. Conduct the update for the codebook.
4. Calculate the quantization distortion value; go to step 2 until the convergence of quantization distortion.

The iterative optimization procedure may not in general yield truly globally optimized quantizers. The algorithm may just yield a locally optimized quantizer. Of course, good initial codebook may enhance the potential of the algorithm to provide better codebooks.

3.2.3 Numerical Data and Discussions

TCVQ achieves better performance compared to TCQ. As the dimension of TCVQ increases, the SDR value increases, in the tradeoff of computational complexity. Tables 3.3 and 3.4 show simulation results for TCVQ for integer and fractional rates, respectively. For an integer rate, the dimension k can be any arbitrary integer (including the scalar case of TCQ). For fractional rates k, kR requires to be an integer. Simulation results [26] on a training sequence of a zero-mean, unit-variance

memoryless Gaussian source of length 1,000,000 samples are presented in Tables 3.3-3.4.

Table 3.3 shows that TCVQ can yield from 0.1 to 0.8 dB improvement in SDR over TCQ. When the dimension is one, the TCVQ reduces to TCQ. Table 3.4 shows that TCVQ with fractional rate has SDR that is higher than that of a vector quantization (VQ) at the same rate by at least 0.5 dB. The 16-state trellis TCVQ with fractional rate is only from 0.38 to 0.58 dB worse than the theoretical rate distortion bound.

TCVQ can also be regarded as a special case as a finite-state vector quantization (FSVQ), which is based on a finite state machine and state-dependent codebook. At each encoding step, TCVQ uses a state-dependent codebook to encode the source vector. Viterbi algorithm is used to search the trellis of the finite state machine. TCVQ achieves better SDR performance than TCQ. While TCQ always have lower computation complexity than TCVQ, TCQ cannot have with fractional rate.

Rate in bits per sample	Dimension of vectors	Trellis size (states)			Theroretical bound
		4	8	16	
1	1	5.00	5.19	5.27	6.02
	2	5.02	5.20	5.29	
	3	5.10	5.22	5.31	
	4	5.13	5.24	5.33	
	6	5.20	5.32	5.42	
2	1	10.53	10.70	10.78	12.04
	2	10.64	10.76	10.84	
	3	10.75	10.85	10.96	
	4	10.89	11.00	11.22	
3	1	16.19	16.33	16.40	18.06
	2	16.34	16.56	16.62	
4	1	21.66	21.78	21.84	24.08
	2	22.32	22.45	22.63	

Table 3.3 The SNR of TCVQ with integer rates for memoryless Gaussian source.

Rate in bits per sample	Trellis size (states)			VQ	Theroretical bound
	4	8	16		
0.500	2.50 (2)	2.59 (2)	2.63 (2)	2.07 (2)	3.01
0.667	3.38 (6)	3.45 (6)	3.51 (6)	2.99 (6)	4.01
0.750	3.80 (4)	3.90 (4)	3.95 (4)	3.35 (4)	4.52
0.800	4.09 (5)	4.18 (5)	4.24 (5)	3.67 (5)	4.82
0.833	4.33 (6)	4.40 (6)	4.45 (6)	3.91 (6)	5.02

Table 3.4 The SNR of TCVQ compared with that of full search VQ and theoretical rate distortion bound for memoryless Gaussian source. The numbers in the parentheses show the dimension of vectors.

Chapter 4
Trellis-Coded Quantization with Unequal Distortion

Although there are many methods for unequal error protection associated with channel coding, the methods for unequal distortion for source coding are usually limited to the concept of information refinement [1] using mostly product or multistage vector quantizers or similar techniques [2]-[3]. The encoding techniques for unequal distortion are usually associated with scalable or multi-resolution signal compression. In practice, most of the previous works provide unequal distortion by changing the encoding rate, i.e. changing the rate R in $D(R) = k2^{-2R}$, where k is a proportional constant depending on quantizer design. The refinement (or multistage) techniques in [1]-[3] provide methods to generate an embedded encoded sequence in which bits for coarse quantization can be reused for fine quantization.

In this thesis, we provide another method for unequal distortion [27] within the same quantizer, i.e.,

$$D_f(R) = k_f 2^{-2R} \quad \text{and} \quad D_c(R) = k_c 2^{-2R} \tag{4.1}$$

where $D_f(R)$ and $D_c(R)$ are the distortion function of fine and coarse quantizers, respectively, and k_f and k_c, $k_f < k_c$, are the corresponding proportional constants. Similar to the case for information refinement [1]-[3], quantizers with unequal distortion are usually worse than the optimal quantizer. In this thesis, the design objective is to ensure that

$$pk_f + (1-p)k_c \leq k \tag{4.2}$$

where $0 \leq p \leq 1$ is the fraction of the signal for fine quantizer. While multistage vector quantizers cannot perform better than single-stage optimal vector quantizers [2], optimal quantizers with unequal distortion cannot perform better than optimal quantizers with equal distortion. The objective of (4.2) intends to guarantee that the average distortion is just a little bit worse than but having no significant difference with the minimum achievable distortion.

As shown in Chapter 3, trellis-coded quantizer (TCQ) [4] has proven to be an efficient quantizer to reduce quantization distortion and trellis-coded vector quantization (TCVQ) is the same as TCQ, with multi-dimensional data sample [24]. In another words, TCQ is a particular case of TCVQ in one dimension. Both TCQ and TCVQ with unequal distortion is designed in this thesis. Usually, for equal distortion, a fixed codebook (or set of quantization levels) is used in the whole TCVQ encoding process. A TCVQ having two different codebooks (two sets of quantization levels) is used here to provide unequal distortion. One of the codebook

has *fine* quantization levels while the other one has *coarse* quantization levels. While the analog samples using fine codebook can provide smaller quantization distortion than that using coarse codebook, the average quantization distortion, like (4.2), is just slightly worse than that with single codebook for equal distortion [27]. Usually, with the optimal design, the fine quantization distortion is also smaller than the minimum achievable distortion for the same encoding rate.

4.1 Design Procedures

The design of unequal distortion TCVQ is based on that of equal distortion one (section 3.2). For the traditional TCVQ, the same codebook is used though out the trellis in the encoding. While for unequal distortion TCVQ, we use different codebooks, with different resolutions, though out the same trellis to provide unequal distortion.

For traditional TCVQ at the rate of R bits/sample, the number of quantization levels is 2^{R+1}. A codebook of 2^{R+1} different codewords is used. For unequal distortion TCVQ at the same rate, each codebook used still have 2^{R+1} codewords but different resolution. In the codebook with lower resolution, half of the codewords are equal to another half and this introduces greater distortion even the codebook size is unchanged. Multi-resolution codebooks of the same size are used for unequal

distortion TCVQ.

For each codebook, codewords are partitioned into subsets according to the algorithm described in section 3.2.1, in order to maximize the minimum distance within the subset.

Finally, the codebooks will be optimized iteratively by the generalized Lloyd algorithm, as the same case in the traditional TCVQ, with equal distortion. Details will be discussed in the following sections.

4.2 Fine and Coarse Codebooks

Using codebooks with different resolutions on the same trellis, we can encode signals with unequal distortion [27]. This can be applied to encode signal to provide better fidelity at some part at the expense of less fidelity at the other part, with the same encoding rate of equal distortion. In general, more than two codebooks could be used to encode different part of the signal with many levels of protection. Without loss of generality, we confine our discussion to the case of only two codebooks.

The codebooks T_0 and T_1 in Figure 4.1 can be used as the fine and coarse codebook, respectively, for unequal distortion. In the coarse codebook T_1, each quantization level represents two different labels in the codebook T_0 and the trellis diagram. For

example, both symbols D_0 and D_1 use the same quantization level. Therefore, using coarse codebook will provide less signal fidelity than the fine codebook.

If TCVQ uses the fine codebook alone, the minimum quantization distortion is equal to optimal TCVQ with equal distortion. If the TCVQ scheme just uses the coarse codebook alone, the performance is the same as 2^R-level Lloyd-Max scalar (or vector) quantizer without trellis structure. The samples using the fine codebook have lower quantization distortion than those using the coarse codebook. The goal of this thesis is to guarantee that the fine distortion is smaller than the minimum achievable distortion for the same data-rate. However, in the same time, the average of fine and coarse distortion is just a little bit larger than the minimum achievable distortion for the same data-rate.

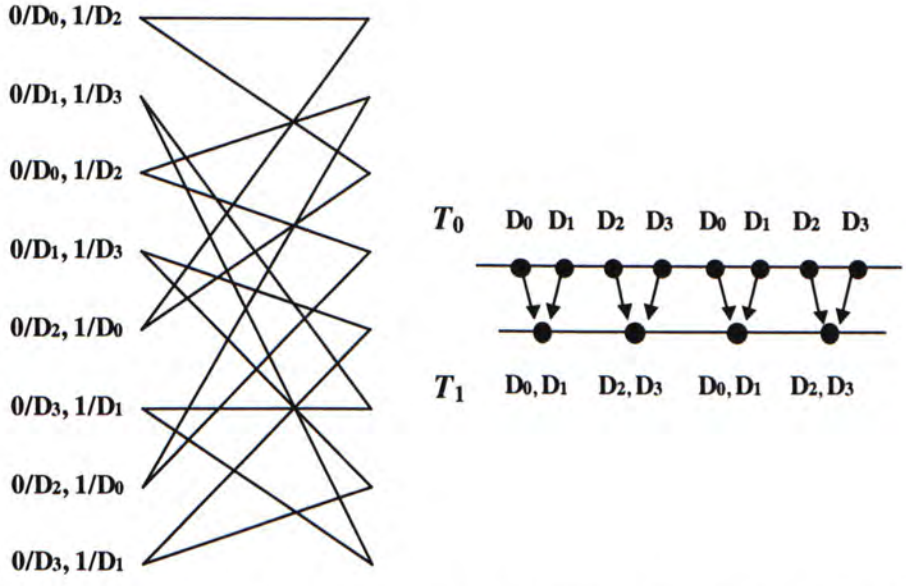


Figure 4.1 An 8-state trellis with subsets labeling and the fine codebook and coarse codebooks, partition for 2 bits/sample scalar TCQ.

The encoding process with unequal distortion is identical to the traditional TCVQ scheme with equal distortion. Because both fine and coarse codebooks have the same trellis structure, Viterbi algorithm is used to trace the trellis diagram to find the sequence with minimum distortion. The discrepancy with conventional TCVQ is the usage of two codebooks in Figure 4.1 instead of one codebook in conventional TCVQ.

Without loss of generality, in latter part of this thesis, we consider the specific case that the fine and coarse codebooks are used alternatively sample after sample. Figure 4.2 shows the 8-state trellis (same as that in Figure 4.1) using fine and coarse codebooks alternatively. For a particular stage, the TCQ coder encodes sample with the fine codebook T_0, it encodes sample with coarse codebook T_1 on the next stage. For example, one fine quantized sample follows by another coarse sample and vice versa. Noted that the unequal distortion scheme can be applied to any permutation and ratio of fine and coarse codebooks. Of course, even distribution of codebooks usually provides better performance for smaller distortion.

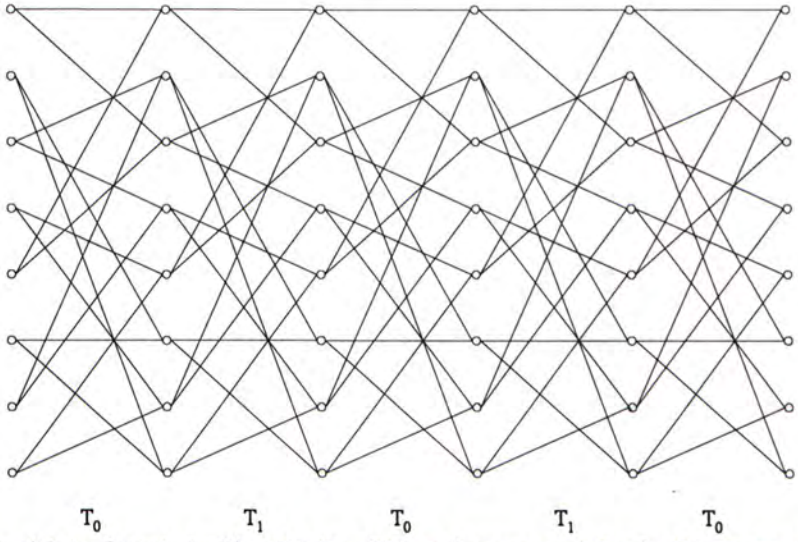


Figure 4.2 An 8-state trellis with fine (T_0) and coarse codebook (T_1) alternatively.

4.3 Set Partitioning

The fine and coarse codebooks are generated by the LBG algorithm [25] as initial codebooks. Afterward, codewords of each codebook are partitioned into subsets. Set partitioning is to maximize the minimum distance within each subset [26]. In this way, free distance of parallel transactions is maximized and quantization distortion can be minimized. As described in section 3.2.1, codewords are partitioned into two subsets, with minimum distance between codewords within subset maximized. The same procedures can be applied on the two subsets to generate four subsets and so on.

4.4 Codebook Optimization

The quantization levels of a codebook for unequal distortion TCVQ can be optimized iteratively using the generalized Lloyd algorithm [2][8][25]. This is similar to the case of equal distortion. Using a training sequence of analog samples, the algorithm is operated as follows:

1. Two initial codebooks with proper set partition, properly the Lloyd-Max quantizers [8] with 2^R and 2^{R+1} levels.
2. Encode the analog training sequence into a sequence of codewords, using both fine and coarse codebooks.
3. Update each codeword with the average of training samples that map to that quantization level, i.e., the centroid of the subset. Conduct the update separately for fine and coarse codebooks.
4. Calculate both fine and coarse quantization distortion values; go to step 2 until the convergence of quantization distortion.

Note that this procedure yield locally optimized codebook. It may not be the globally optimized one. The LBG algorithm can usually provide good initial codebook.

Simulations are performed on memoryless Gaussian source and Markov Gaussian source in chapter 5 and 6, respectively. Numerical data and discussions will be presented in details in the following chapters.

4.5 Encoding for Unequal Distortion TCVQ

Viterbi algorithm is used to encode unequal distortion TCVQ [27]. The Viterbi algorithm finds the path metric by sequentially moving the trellis stage by stage [15][12]. We have discussed decoding TCQ with Viterbi algorithm based on trellis representation on TCM encoder in section 2.2.5.

Chapter 5
Unequal Distortion TCVQ on Memoryless Gaussian Source

In chapter 4, a trellis-coded vector quantization (TCVQ) encoder having two unequal distortion values is proposed for encoding signals. In this chapter, we apply the TCVQ encoder for memoryless Gaussian source. The trellis diagram and codebooks are similar to that in Figure 4.1. The fine and coarse codebooks are optimized, using the procedure in section 4.4. Using Viterbi algorithm, the encoding process of unequal distortion TCVQ is identical to that with equal distortion, except that different codebooks are used alternatively. The only difference is the metric in the Viterbi algorithm using different quantization codebooks. The computation complexity is also the same as that with equal distortion.

Figure 5.1 shows an example scalar TCQ with unequal distortion at the rate of 2 bits per sample, and the fine and coarse codebooks used for encoding [12]. The two codebooks contribute to encoding with different distortions. Figure 5.2 shows encoding path of the trellis of the unequal distortion TCQ. Note that the TCQ coder encodes with the fine codebook T_0 at the first stage, and coarse codebook T_1 at the second stage, and so on alternatively. Distortions are measured from the simulations

on memoryless Gaussian source and will be presented in the later sections.

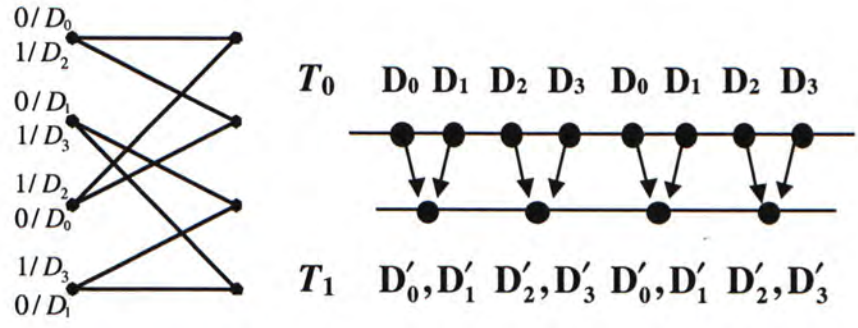


Figure 5.1 Ungerboeck's four-state amplitude modulation trellis with subsets labeling and the fine codebook and coarse codebook, partition for 2 bits/sample scalar TCQ.

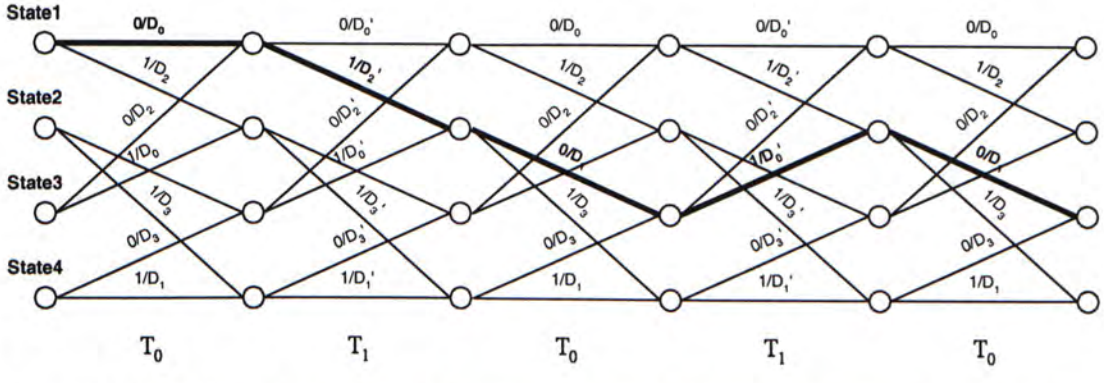


Figure 5.2 A 4-state TCQ with fine and coarse codebooks, with path sequence $\{u\}=\{0,1,0,1,0\}$ and output symbol sequence $\{v\}=\{D_0,D'_2,D_1,D'_0,D_1\}$.

5.1 Memoryless Gaussian Source

Gaussian memoryless/independent source $\{x_k\}$ is a sequence of random variable x. The probability density function of x is given by:

$$f_x(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(x-m_x)^2}{2\sigma^2}} \tag{5.1}$$

where m_x is the mean and σ^2 is the variance of the random variable x. In our simulated system, a sequence of Gaussian random variable with zero mean and unit variance is used as input samples.

5.2 Set Partitioning of Codewords of Memoryless Gaussian Source

Set partitioning of codewords is to maximize the minimum distance between codewords within subsets [26]. Using memoryless Gaussian source as coding signal, and grouping 2 samples into 2-dimensional vector, codebook is designed by the LBG algorithm and then set partitioned into subsets as described in section 3.2.1.

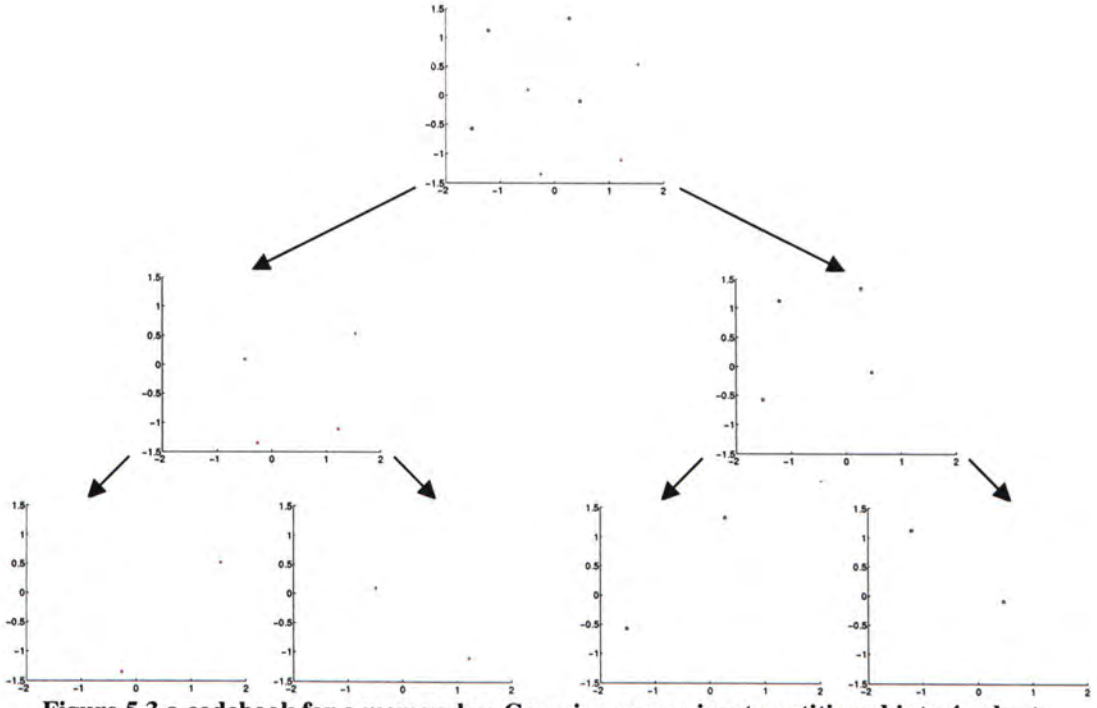


Figure 5.3 a codebook for a memoryless Gaussian source is set partitioned into 4 subsets.

For a memoryless Gaussian source with 2 samples grouped as a 2-dimensional vector, Figures 5.3 shows partitioning on the irregular codebook of 16 codewords of the source into 2 subsets and 4 subsets, respectively. Using the partitioning algorithm, the closest 2 codewords are always assigned into different subsets. When two subsets are generated, the algorithm can be applied to these two subsets to generate 4 subsets.

5.3 Numerical Results and Discussions

Training sequence of length of 100,000 samples is used in the simulations. Tables 5.1-5.3 show the signal-to-distortion ratio (SDR) of scalar TCQ, 2-dimesional TCVQ, and 3-dimensional TCVQ, respectively, for memoryless Gaussian source [27]. All values of SDR are measured in unit of dB, and defined as

$$SDR = \frac{\sum_{k=1}^{K} x_k^2}{\sum_{k=1}^{K} (x_k - \hat{x}_k)^2}, \tag{5.2}$$

where $\hat{x}_k$ is the reproductive codeword for the signal sequence x_k of length K. The simulations are performed at various rates and trellis with various numbers of states. The fine codebook T_0 and the coarse codebook T_1 are used for unequal distortion. Two different distortion values are calculated separately for fine and coarse quantization. With equal distortion, the optimal performance with maximum SDR from [4] is also shown for comparison. Similar to [4], the trellises listed in [13] are used in scalar TCQ for Table 5.1.

From Tables 5.1-5.3, SDR of fine and coarse quantization is always better and worse than the SDR for equal distortion, respectively. The SDR of coarse quantization is worse than that of Lloyd-Max quantizer with the same coding rate. In general, SDR of both fine and coarse quantization increases with both coding rate and the number of states. Tables 5.1-5.3 also show the average SDR of both fine and coarse SDR.

The average SDR of the fine and coarse quantization is calculated by finding the average quantization distortion, similar to (4.2), and then divided by the signal power. The average SDR of the fine and coarse distortion is simply called the *average distortion* in the later paragraphs. The average distortion is slightly worse than the SDR of equal distortion.

For scalar TCQ, the SDR of fine distortion is 0.5 to 0.9 dB higher than that with equal and minimum distortion. For 2-dimensional TCVQ, the SDR of fine distortion is also 0.5 to 0.9 dB higher than that with equal distortion. For 3-dimensional TCVQ, the SDR of fine distortion is 0.5 to 0.6 dB higher than that with equal distortion. It is shown that the SDR of fine quantization is always better than the SDR for equal distortion.

For scalar TCQ, the average SDR is 0.2 to 0.9 dB lower than that with the equal and minimum distortion. For 2-dimensional TCVQ, the average SDR is 0.2 to 0.6 dB lower than that with equal distortion. For 3-dimensional TCVQ, the average SDR is 0.2 to 0.4 dB lower than that with equal distortion. It is shown that the average SDR is close to that with equal distortion. Usually, the discrepancy between the average of unequal distortion and the equal distortion increases with rate. Also, the discrepancy decreases as the dimension increases.

Depending on coding rate and number of states, the SDR difference between the fine and coarse quantization is from 1.5 to 2.8 dB, 1.6 to 2.2 dB and 1.4 to 1.5 dB for

scalar TCQ, 2-D TCVQ and 3-D TCVQ, respectively. Usually, the difference in SDR increases with rate.

Figures 5.4-5.6 plot the SDR as a function of code rate for scalar TCQ, 2-dimensional TCVQ and 3-dimensional TCVQ, respectively, for 16-state trellis and memoryless Gaussian source. These figures confirm that the average SDR is very close to the minimum achievable SDR for equal distortion, i.e., expression equation (4.2).

Average SDR increases when the dimension of TCVQ increases, that means from scalar TCQ to 2-D TCVQ and 3-D TCVQ. Comparing Tables 5.1-5.3, the average SDR increases 0.3 and 0.5 dB when it changes from scalar TCQ to 2-D and 3-D TCVQ, at the rate 2 bits/sample and 16-state trellis, for memoryless Gaussian source. Similar to vector quantization, the higher dimension is the unequal distortion TCVQ, the better is the performance. Note that higher dimension always means higher complexity.

Rate in bits per sample	no. of states	Equal Distortion	Unequal Distortion		
			fine	coarse	average
1	4	5.04	5.57	4.03	4.73
	8	5.19	5.93	4.16	4.96
	16	5.23	5.95	4.20	4.99
2	4	10.53	11.06	8.81	9.81
	8	10.68	11.41	9.05	10.08
	16	10.74	11.36	9.12	10.09
3	4	16.14	16.69	14.15	15.26
	8	16.37	17.18	14.35	15.55
	16	16.42	17.08	14.48	15.59
4	4	21.67	22.27	19.91	20.92
	8	21.78	22.72	19.89	21.07
	16	21.92	22.66	20.02	21.13

Table 5.1 SDR (in dB) of equal and unequal distortion scalar TCQ for memoryless Gaussian source.

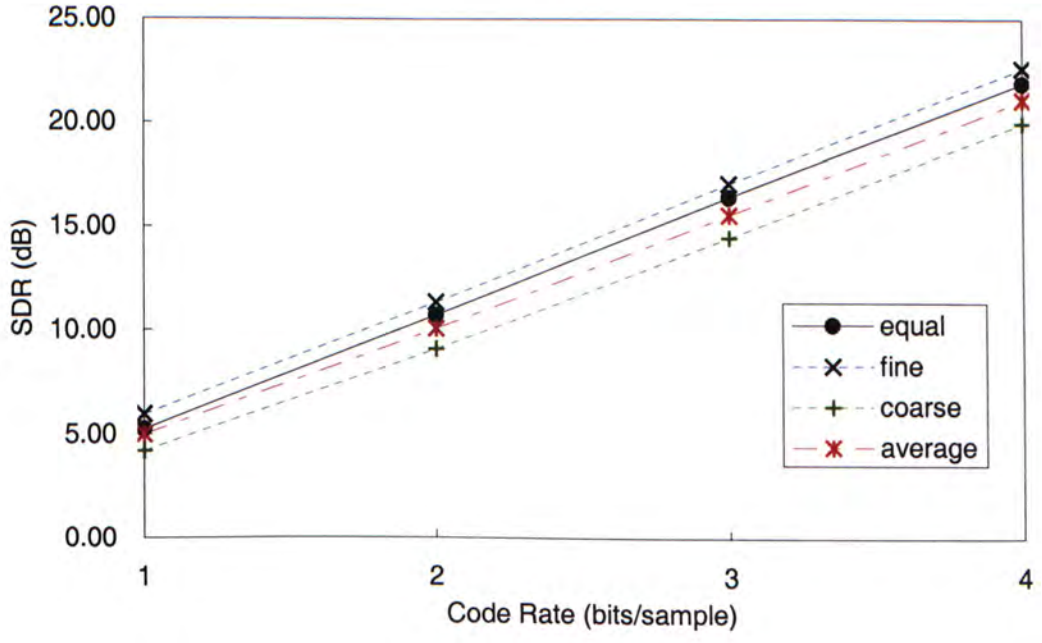


Figure 5.4 SDR performance as a function of code rate for 16-state scalar TCQ for memoryless Gaussian source.

Rate in bits per sample	no. of states	Equal Distortion	Unequal Distortion		
			fine	coarse	average
1	4	5.09	5.63	4.08	4.78
	8	5.20	5.93	4.19	4.97
	16	5.26	5.88	4.24	4.99
2	4	10.68	11.30	9.56	10.34
	8	10.80	11.17	9.60	10.31
	16	10.84	11.37	9.61	10.40
3	4	16.49	16.96	15.11	15.93
	8	16.59	17.24	15.24	16.11
	16	16.64	17.18	15.32	16.14
4	4	22.74	23.51	21.40	22.32
	8	22.83	23.72	21.48	22.45
	16	22.88	23.74	21.54	22.50

Table 5.2 SDR (in dB) of equal and unequal distortion 2-dimensional TCVQ for memoryless Gaussian source.

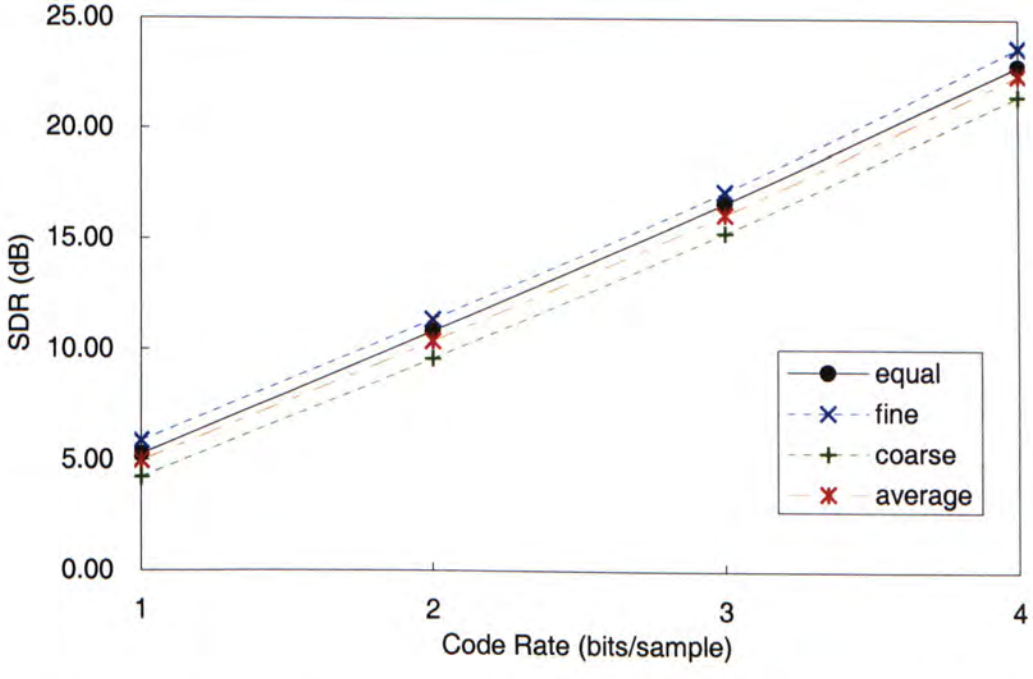


Figure 5.5 SDR performance as a function of code rate for 16-state 2-dimensional TCVQ for memoryless Gaussian source.

rate in bits per sample	no. of states	Equal Quantization	Unequal Quantization		
			fine	coarse	overall
1	4	5.14	5.60	4.23	4.86
	8	5.20	5.70	4.33	4.96
	16	5.28	5.71	4.38	4.99
2	4	10.83	11.40	9.90	10.59
	8	10.91	11.31	9.91	10.56
	16	10.93	11.43	9.91	10.61

Table 5.3 SDR (in dB) of equal and unequal distortion 3-dimensional TCVQ for memoryless Gaussian source.

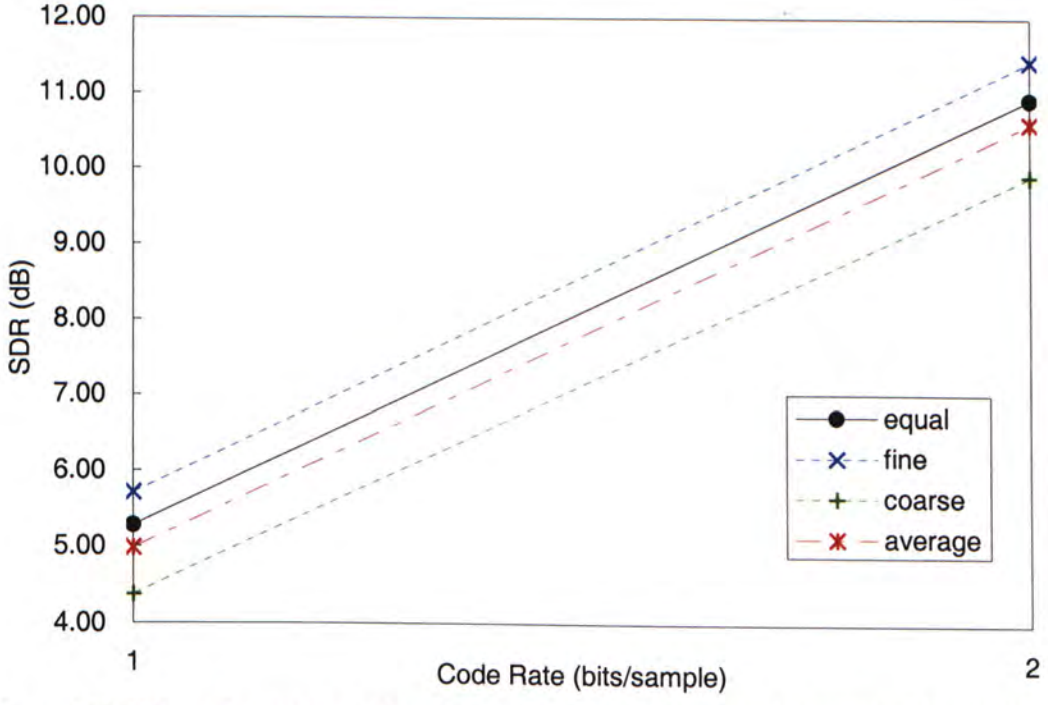


Figure 5.6 SDR performance as a function of code rate for 16-state 3-dimensional TCVQ for memoryless Gaussian source.

Chapter 6
Unequal Distortion TCVQ on Markov Gaussian Source

In the previous chapter, a TCVQ encoder having two unequal distortion values is designed for memoryless Gaussian source. In this chapter, the coder is applied to Markov Gaussian source. Same algorithm and design procedures are used as mentioned in chapter 5, with different training samples.

Although many real-life signals can be modeled as Gaussian distributed random variable, they are in general correlated or having dependence from sample to sample. Therefore, TCVQ are applied to the Markov Gaussian source.

6.1 Markov Gaussian Source

A Markov/correlated Gaussian source is the classic model for correlated samples of speech and images. The theory for Markov source is straightforward and can describe practical source well. A Gaussian source is *autoregressive of order k* if it satisfies

$$x_j = \sum_{i=1}^{k} a_i x_{j-i} + u_j \tag{6.1}$$

where the u_j are zero-mean independent Gaussian with variance σ^2. It is a filtered memoryless Gaussian source where the filter is the all-pole with z-transform $1/(1-A(z))$, where $A(z) = a_1 z^{-1} + a_2 z^{-2} + \cdots + a_k z^{-k}$. The Markov Gaussian sources provide a very simple model for real-life signals. Speech, images, biowaveforms, and solar signals all have a low pass spectrum. Low order all-pole filters are the simplest realization of a low pass filter. It is natural to model these signals as the output of such a filter with an independent sample excitation at the input. This is particular close to the model for speech generation. The excitation is white noise for unvoiced sounds and a pulse train for voiced sounds, both of which can be viewed as white in spectrum, while the all-pole filter coefficients can be related directly to vocal tract parameters such as the spacing of the lips, teeth and palate.

6.2 Set Partitioning of Codewords of Markov Gaussian Source

The initial codebook is generated by the LBG algorithm and then set partition the codewords into subsets as the procedures in section 3.2.1. With 2-dimensional 1st order Markov Gaussian source of correlated coefficient ρ = 0.9 as training sequence, codebook with 32 codewords is generated and set partitioned into 2 and 4 subsets as

shown in Figures 6.1.

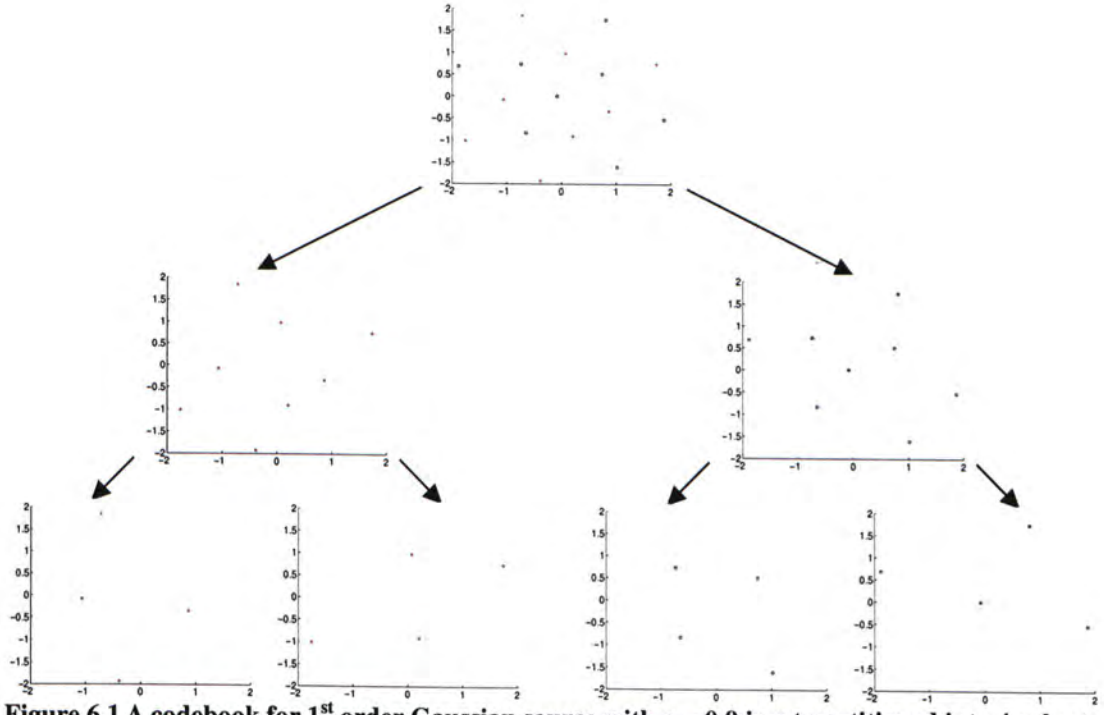


Figure 6.1 A codebook for 1st order Gaussian source with ρ = 0.9 is set partitioned into 4 subsets.

6.3 Numerical Results and Discussions

The simulations of unequal distortion TCVQ are performed on 1st order Markov Gaussian source, with correlation coefficient, ρ = 0.9. Tables 6.1-6.3 show the SDR of scalar TCQ, 2- and 3-dimensional TCVQ, respectively, for 1st order Markov Gaussian source [27].

From Tables 6.1-6.3, SDR with fine and coarse quantization is always better and worse than the SDR for equal distortion, respectively. The SDR of coarse quantization is even worse than that of Lloyd-Max quantizer with the same coding rate. In general, SDR of both fine and coarse quantization increases with both coding rate and the number of states. Tables 6.1-6.3 also show the average SDR of both fine and coarse SDR.

For scalar TCQ, the average SDR is up to 1.0 dB higher than that of the equal distortion. For 2-dimensional TCVQ, the average SDR is up to 0.9 dB higher than that of the equal distortion. For 3-dimensional TCVQ, the average SDR is up to 0.6 dB higher than that of the equal distortion. It is show that the SDR of fine quantization is better than that of equal distortion.

From Table 6.1, for scalar TCQ, the average SDR is 0.5 to 1.3 dB lower than that with equal distortion. From Table 6.2, for 2-dimensional TCVQ, the average SDR is 0.3 to 0.6 dB lower than that with equal distortion. From Table 6.3, for 3-dimensional TCVQ, the average SDR is 0.2 to 0.3 dB lower than that with equal distortion. The average SDR is quite close to that of equal distortion, in particular for high dimension TCVQ. Usually, the discrepancy between the average distortion and the equal distortion increases with rate, similar to memoryless Gaussian source. Also, the discrepancy decreases as the dimension increases.

Depending on coding rate and number of states, the SDR difference between the fine

and coarse quantization is from 1.5 to 2.6 dB, 1.3 to 2.2 dB and 1.1 to 1.5 dB for scalar TCQ, 2-D TCVQ and 3-D TCVQ, respectively. Usually, the difference in SDR increases with rate.

Figures 6.2-6.4 plot the SDR as a function of code rate for scalar TCQ, 2-dimensional TCVQ and 3-dimensional TCVQ, respectively, for 16-state trellis and Markov Gaussian source. Figures 6.2-6.4 show that the average SDR is close to the minimum achievable SDR for equal distortion, similar to the case of memoryless source.

Average SDR increases when the dimension of TCVQ increases. Comparing Tables 6.1-6.3, the average of unequal distortion increases 3.8 and 5.2 dB when it changes from scalar TCQ to 2- and 3-dimensional TCVQ, at the rate of 2 bits/sample and 16-state trellis, for 1st order Markov Gaussian source. The changes are much more significant in the case of Markov Gaussian source, compared to memoryless Gaussian source.

Comparing Tables 5.1 and 6.1, the average SDR of scalar TCQ of Markov Gaussian source is up to 0.3 dB higher than that of memoryless source. If 2-dimensional TCVQ is applied, the difference is much higher. The average SDR of 2-dimensional TCVQ of Markov source is 3.4 to 3.8 dB higher than that of memoryless source. The difference of up to 4.9 dB is even higher for 3-dimensional TCVQ.

Rate in bits per sample	no. of states	Equal Distortion	Unequal Distortion		
			fine	coarse	average
1	4	5.35	5.66	4.15	4.84
	8	6.46	6.48	4.22	5.20
	16	6.58	6.59	4.32	5.30
2	4	10.58	10.95	9.22	10.00
	8	10.95	11.44	9.21	10.18
	16	11.02	11.56	9.34	10.31
3	4	15.78	16.24	14.39	15.22
	8	16.19	17.00	14.48	15.57
	16	16.35	17.24	14.61	15.73
4	4	21.44	22.10	20.02	20.93
	8	21.74	22.76	20.15	21.26
	16	21.98	22.92	20.29	21.41

Table 6.1 SDR (in dB) of equal and unequal distortion scalar TCQ for Markov Gaussian source with $\rho = 0.9$

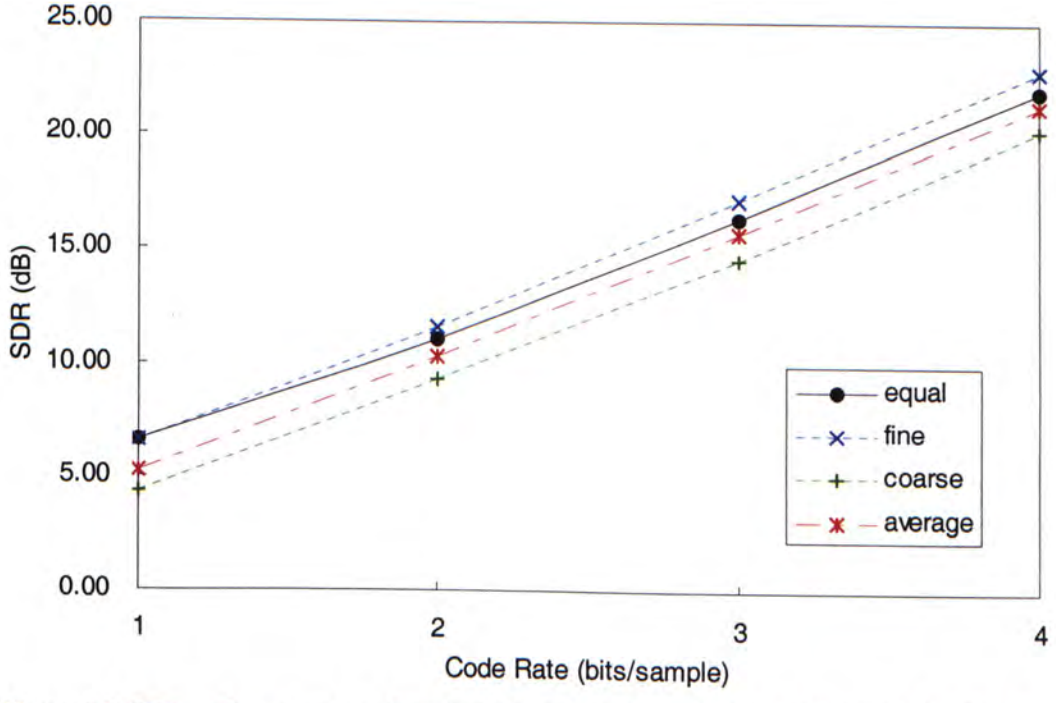


Figure 6.2 SDR performance as a function of code rate for 16-state scalar TCQ for Markov Gaussian source with $\rho = 0.9$

Rate in bits per sample	no. of states	Equal Distortion	Unequal Distortion		
			fine	coarse	average
1	4	8.80	8.86	7.59	8.18
	8	8.87	9.17	7.69	8.37
	16	8.94	9.32	7.81	8.50
2	4	14.28	14.51	13.26	13.84
	8	14.39	14.98	13.34	14.08
	16	14.45	14.94	13.43	14.12
3	4	20.08	20.54	18.80	19.58
	8	20.19	20.91	18.92	19.80
	16	20.26	20.91	18.95	19.82
4	4	26.24	26.88	24.98	25.83
	8	26.37	27.23	25.05	26.00
	16	26.42	27.22	25.12	26.04

Table 6.2 SDR (in dB) of equal and unequal distortion 2-dimensional TCVQ for Markov Gaussian source with $\rho = 0.9$

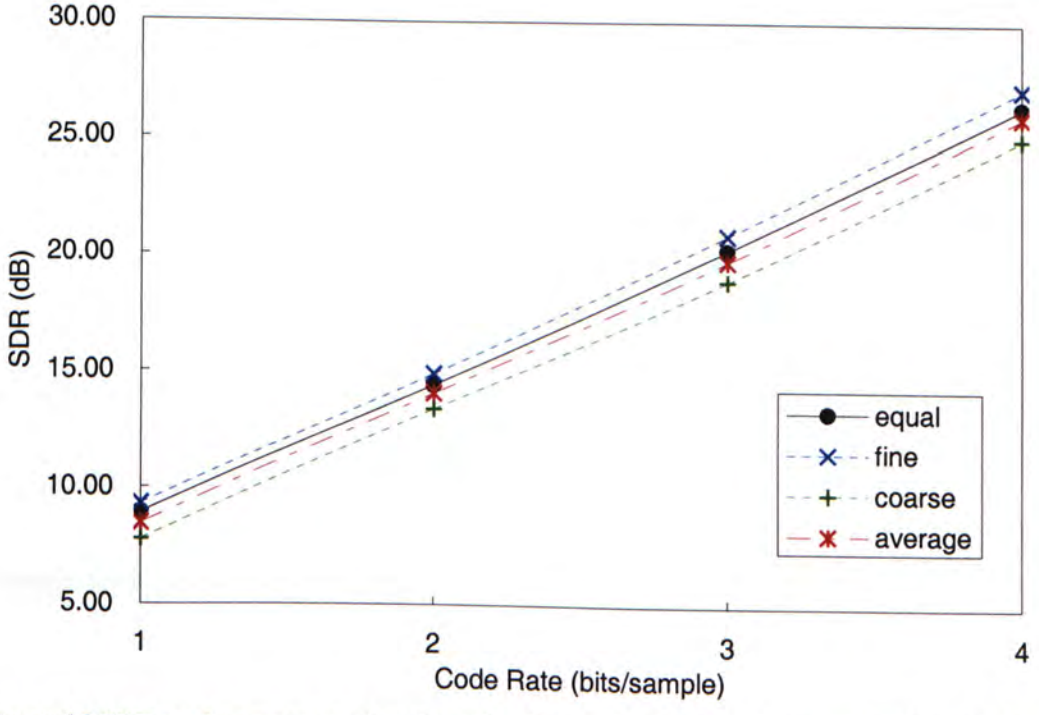


Figure 6.3 SDR performance as a function of code rate for 16-state 2-dimensional TCVQ for Markov Gaussian source with $\rho = 0.9$

rate in bits per sample	no. of states	Equal Quantization	Unequal Quantization		
			fine	coarse	overall
1	4	10.03	10.31	9.21	9.73
	8	10.08	10.57	9.25	9.86
	16	10.12	10.53	9.32	9.88
2	4	15.63	16.08	14.73	15.35
	8	15.72	16.28	14.81	15.48
	16	15.76	16.23	14.87	15.50

Table 6.3 SDR (in dB) of equal and unequal distortion 3-dimensional TCVQ for Markov Gaussian source with $\rho = 0.9$

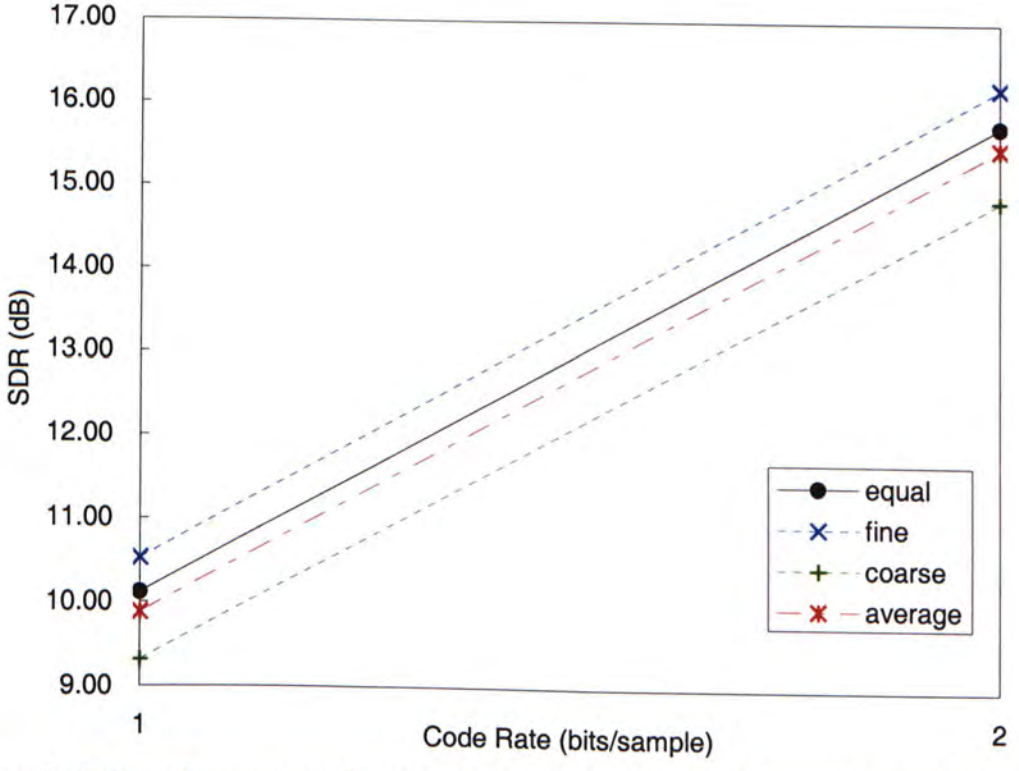


Figure 6.4 SDR performance as a function of code rate for 16-state 3-dimensional TCVQ for Markov Gaussian source with $\rho = 0.9$

We performed the simulations on Markov Gaussian source again, with correlation coefficient, ρ = 0.5. Tables 6.4-6.6 show the SDR of scalar TCQ, 2-dimensional TCVQ and 3-dimensional TCVQ, respectively. Figures 6.5-6.7 plot the SDR as a function of code rate for scalar TCQ, 2-dimensional TCVQ and 3-dimensional TCVQ, respectively, for 16-state trellis. As an intermediate between memoryless source and correlated source with ρ = 0.9, there are continuous trends in various aspects.

For scalar TCQ, the SDR of fine distortion is 0.5 to 0.9 dB higher than that with equal and minimum distortion. For 2-dimensional TCVQ, the SDR of fine distortion is 0.3 to 1.2 dB higher than that with equal distortion. For 3-dimensional TCVQ, the SDR of fine distortion is 0.3 to 0.5 dB higher than that with equal distortion. It is shown that the SDR of fine quantization is always better than the SDR for equal distortion.

From Table 6.4, for scalar TCQ, the average SDR is 0.2 to 0.8 dB lower than that with equal distortion. From Table 6.5, for 2-dimensional TCVQ, the average SDR is 0.2 to 0.5 dB lower than that with equal distortion. While from Table 6.6, for 3-dimensional TCVQ, the average SDR is 0.2 to 0.3 dB lower than that with equal distortion. Both the value and discrepancy decreases as the dimension increases. These figures show the average of unequal distortion is close to that of equal distortion.

Average SDR increases when the correlation of the Gaussian source increases. From Tables 5.1 and 6.4, comparing memoryless Gaussian source and Markov Gaussian source with correlation coefficient ρ = 0.5, the average SDR is 0.1 dB higher, at the rate of 2 bits/sample and for 16-state trellis. From Tables 5.1 and 6.1, comparing memoryless Gaussian source and Markov Gaussian source with correlation coefficient ρ = 0.9, the average SDR is 0.2 dB higher, at the same rate and same trellis. This shows that higher the correlation of the sample source, the higher average SDR can be achieved by unequal distortion scalar TCQ. This result is more obvious in the case of 2-dimensional TCVQ. From Tables 5.2 and 6.5, comparing memoryless Gaussian source and Markov Gaussian source with correlation coefficient ρ = 0.5, the average SDR is 0.6 dB higher. From Tables 5.2 and 6.2, comparing memoryless Gaussian source and Markov Gaussian source with correlation coefficient ρ = 0.9, the average SDR is 3.7 dB higher. The average SDR of Markov source with correlation coefficient ρ = 0.5 is 1.1 dB higher than that of memoryless source, and that of Markov source with correlation coefficient ρ = 0.9 is 4.9 dB higher than that of memoryless source.

rate in bits per sample	no. of states	Equal Quantization	Unequal Quantization		
			fine	coarse	overall
1	4	5.17	5.70	4.00	4.77
	8	5.20	5.99	4.13	4.96
	16	5.28	6.09	4.25	5.07
2	4	10.55	11.08	8.97	9.90
	8	10.71	11.60	9.04	10.13
	16	10.81	11.45	9.25	10.20
3	4	16.23	16.71	14.34	15.37
	8	16.39	17.16	14.46	15.59
	16	16.51	17.29	14.52	15.69
4	4	21.65	22.27	19.88	20.92
	8	21.97	22.88	19.99	21.21
	16	22.21	22.97	20.21	21.36

Table 6.4 SDR (in dB) of equal and unequal distortion scalar TCQ for Markov Gaussian source with $\rho = 0.5$

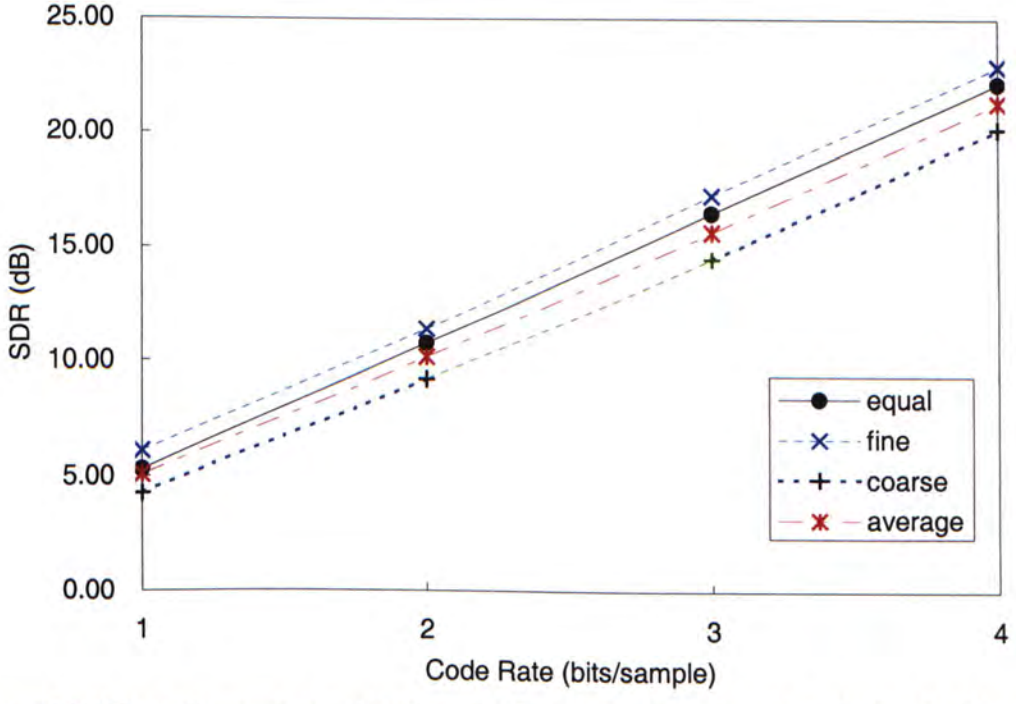


Figure 6.5 SDR performance as a function of code rate for 16-state scalar TCQ for Markov Gaussian source with $\rho = 0.5$

rate in bits per sample	no. of states	Equal Quantization	Unequal Quantization		
			fine	coarse	overall
1	4	5.77	6.05	4.90	5.43
	8	5.87	6.40	4.98	5.62
	16	5.92	6.40	5.03	5.65
2	4	11.23	11.62	10.06	10.76
	8	11.36	11.93	10.14	10.94
	16	11.42	11.88	10.22	10.97
3	4	17.11	17.60	15.76	16.58
	8	17.23	17.86	15.84	16.72
	16	17.27	17.84	15.90	16.75
4	4	23.69	24.66	22.21	23.30
	8	23.81	24.96	22.30	23.46
	16	23.83	24.91	22.38	23.49

Table 6.5 SDR (in dB) of equal and unequal distortion 2-dimensional TCVQ for Markov Gaussian source with $\rho = 0.5$

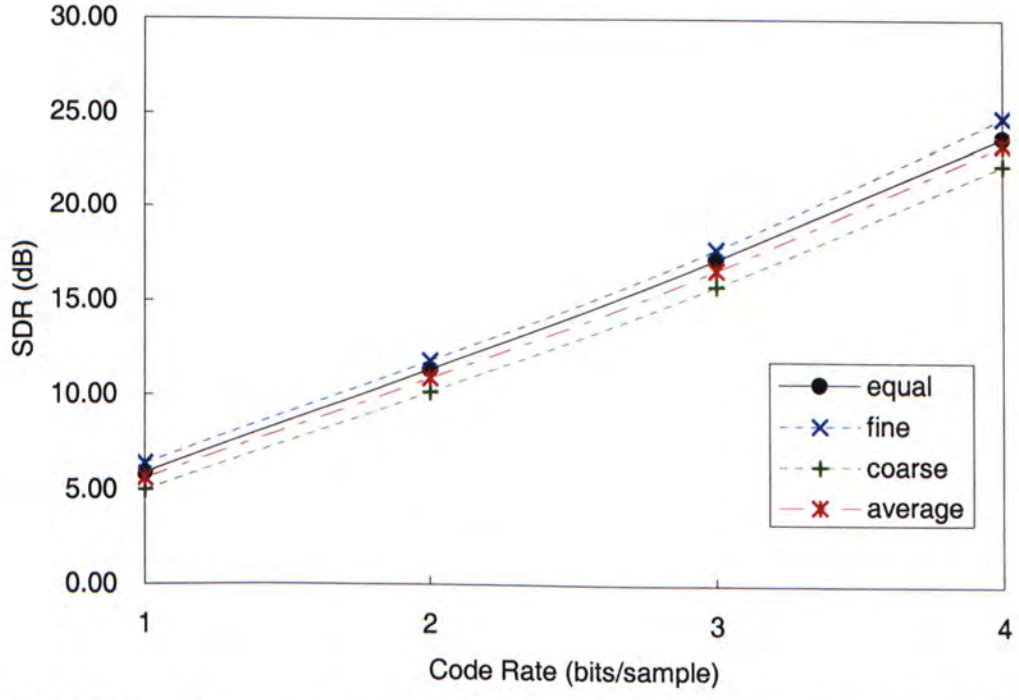


Figure 6.6 SDR performance as a function of code rate for 16-state 2-dimensional TCVQ for Markov Gaussian source with $\rho = 0.5$

rate in bits per sample	no. of states	Equal Quantization	Unequal Quantization		
			fine	coarse	overall
1	4	6.23	6.54	5.41	5.93
	8	6.31	6.75	5.50	6.07
	16	6.34	6.69	5.57	6.09
2	4	11.83	12.20	10.94	11.53
	8	11.92	12.39	11.06	11.67
	16	11.95	12.39	11.10	11.70

Table 6.6 SDR (in dB) of equal and unequal distortion 3-dimensional TCVQ for Markov Gaussian source with ρ = 0.5

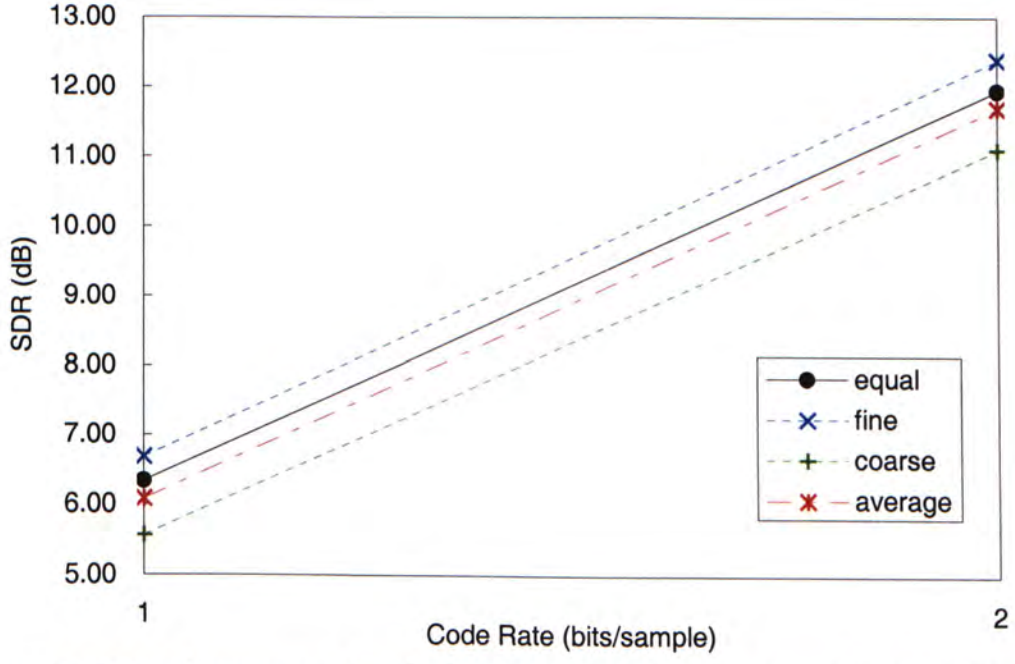


Figure 6.7 SDR performance as a function of code rate for 16-state 3-dimensional TCVQ for Markov Gaussian source with ρ = 0.5

Chapter 7

Conclusions

Trellis-coded quantizers using two different codebooks are designed for unequal distortion. The computational complexity is the same as that with one set of codebook for equal distortion. While samples quantized with fine and coarse codebooks achieve better and worse distortion, respectively, than that with single codebook. The average SDR of the fine and coarse distortion (average SDR) is slightly less than the minimum achievable distortion.

For 2-dimensional TCVQ, the SDR of the fine distortion is up to 0.9 dB higher than that of the equal distortion, for both memoryless and Markov Gaussian (ρ=0.9) sources. The fine distortion is always better than that of equal distortion.

For 2-dimensional TCVQ, the average SDR of unequal distortion is 0.2-0.6 dB and 0.3-0.6 dB lower than that with equal distortion, for memoryless and Markov Gaussian sources (ρ=0.9), respectively. The performance of unequal distortion coder is close to the equal distortion one.

Performance is always improved with the increase of the dimension of TCVQ is applied. At the rate of 2 bits/sample and 16-state trellis, the increase in average SDR for 2-dimensional TCVQ over scalar TCQ is 0.3 and 3.8 dB, for memoryless and Markov Gaussian (ρ=0.9) sources, respectively.

The unequal distortion TCVQ coder performs better in correlated sources than memoryless source. The average SDR of Markov Gaussian sources (ρ=0.9) is up to 0.3 dB higher than that of memoryless Gaussian source, for scalar TCQ. The difference is much higher if 2-dimensional TCVQ is used, resulting 3.4-3.8 dB improvements. For 3-dimensional TCVQ, the average distortion of Markov Gaussian source is up to 4.9 dB higher than that of the memoryless one.

Future work can be done on applying unequal distortion TCVQ over signal compression. TCVQ with unequal distortion can be applied for systems with two different classes of signal, each having different importance. In image compression using wavelet transform, as a simple example, the fine distortion TCVQ can be applied to low-frequency subbands having higher visual importance and the coarse quantization can be applied to high-frequency subbands having lower visual importance. Although this kind of scalable signal compression can be traditionally accomplished through bit allocation, quantizers with unequal distortion may provide better method to resolve this problem.

Bibliography

[1] W. H. R. Equitz and T. M. Cover, "Successive Refinement of Information," *IEEE Trans. Inform. Theory*, vol. 37, pp. 269-275, March 1991.

[2] A. Gresho and R. M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic: Boston, 1992.

[3] H. Jafarkhani and V. Tarokin, "Design of Successively Refinable Trellis-Coded Quantizers," *IEEE Trans. Inform. Theory*, vol. 45, pp. 1490-1497, July 1999.

[4] M. W. Marcellin and T. R. Fischer, "Trellis Coded Quantization of Memoryless and Gauss-Markov Sources," *IEEE Trans. Commun.*, vol. 38, pp. 82-93, Jan. 1990.

[5] J. Max, "Quantizing for Minimum Distortion," *IRE Transactions on Information Theory*, vol. IT-6, pp. 7-12, March 1960.

[6] T. M. Cover and A. Joy, *Elements of Information Theory*, Wiley, 1991.

[7] S. P. Lloyd, "Least Squares Quantization in PCM," *IEEE Transactions on Information Theory*, vol. IT-28, pp.129-137, March 1982.

[8] R. M. Gray "Vector Quantization," *IEEE ASSP Mag.*, vol. 1, pp. 4-29, April 1984.

[9] S. Haykin, *Digital Communications*, New York: Wiley, 1998.

[10] G. Ungerboeck, "Channel Coding with Multilevel/Phase Signals," *IEEE Trans. Inform. Theory*, vol. IT-28, pp. 55-67, Jan. 1982.

[11] L. Shu and D. J. Constello Jr., *Error Control Coding: Fundamentals and Applications*, Prentice-Hall, 1983.

[12] C. Schlegel, *Trellis Coding*, New York: IEEE Press, 1997.

[13] E. Biglieri, D. Divsalar, P. J. McLane, and M. K. Simon, *Introduction to Trellis-Coded Modulation with Applications*. New York: Macmillian, 1991.

[14] C. Hartmann and L. Rudolph, "An Optimum Symbol-by-symbol Decoding Rule For Minimizing Symbol Error Rate," *IEEE Transactions on Information Theory*, vol. 20, pp. 284-287, March 1974.

[15] A. J. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," *IEEE Transactions on Information Theory*, IT-13, vol. 2, pp. 260-269, April 1967.

[16] B. Belzer, John D. Villasenor and B. Girod, "Joint Source Channel Coding of Images with Trellis Coded Quantization and Convolutional Codes", *Proceedings of the 1995 IEEE International Conference on Image Processing (ICIP'95)*, vol. 2, pp. 85-88, Oct. 1995.

[17] P. J. Sementilli, A. Bilgin, F. Sheng, M. W. Marcellin, and J. C. Kieffer, "Progressive Transmission in Trellis Coded Quantization-based Image Coders," *International Conference on Image Processing*, vol. 1, pp. 588 –591, 1997.

[18] R. L. Joshi, V. J. Crump, and T. R. Fischer, "Image Subband Coding Using Arithmetic Coded Trellis Coded Quantization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, no. 6, pp. 515-523, Dec. 1995.

[19] J. H. Kasner, M. W. Marcellin and B.R. Hunt, "Universal TCQ in Wavelet Image Coding," *Thirtieth Asilomar Conference on Signals, Systems and Computers*, vol.2, pp. 1279 –1283, 1997.

[20] M. W. Marcellin, T. R. Fischer and J. D. Gibson, "Predictive Trellis Coded Quantization of Speech," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 38, No. 1, pp. 46-54, January 1990.

[21] H. Brunk and N. Farvardin, "Embedded Trellis Coded Quantization," *Data Compression Conference*, pp. 93 –102, 1998.

[22] Aksu and M. Salehi, "Multistage Trellis Coded Quantisation (MS-TCQ) Design and Performance," *IEEE Proc.-Communications*, vol. 144, no. 2, pp. 61-63, April 1997.

[23] C. E. Shannon, "A Mathematical Theory of Communication," *Bell System Technical Journal*, vol 27, pp. 379-423, July 1948.

[24] T. R. Fischer, M. W. Marcellin, and M. Wang, "Trellis-Coded Vector Quantization," *IEEE Transactions on Information Theory*, vol. 37, no. 6, pp.1551-1566, Nov. 1991.

[25] Y. Linde, A. Buzo, and R. Gary "An Algorithm for Vector Quantizer Design," *IEEE Trans. Commun.*, vol. COM-28, no. 1, pp. 84-95, Jan 1980.

[26] S. W. Hong and N. Moayeri, "Trellis Coded Vector Quantization", *IEEE Trans. Commun.*, vol. 40, no. 8, pp. 1273-1276, Aug. 1992.

[27] C. F. Kwong, K. P. Ho and K. H. Chei, "Trellis-Coded Quantization with Unequal Distortion," *Asia Pacific Conference in Commun.*, pp. 980-984, Nov. 2000.