Improving the Error Resilience of G.711.1 Speech Coder with Multiple Description Coding

by

Hooman Alikhanian

A thesis submitted to the Electrical and Computer Engineering Department in conformity with the requirements for the degree of Master of Applied Science

Queen’s University
Kingston, Ontario, Canada
June 2010

Copyright © Hooman Alikhanian, 2010
Abstract

This thesis devises quantization and source-channel coding schemes to increase the error robustness of the newly standardized ITU-T G.711.1 speech coder. The schemes employ Gaussian mixture model (GMM) based multiple description quantizers (MDQ). The thesis reviews the literature focusing on GMM based quantization, MDQ, and GMM-MDQ design methods and bit allocation schemes. GMM-MDQ are then designed for the quantization and coding of the MDCT coefficients in the G.711.1 speech coder. The designs are optimized for and tested over packet erasure channels. Performance of the designs are compared with Mohr’s forward error correcting code based multiple description coding (MDC) scheme.
Acknowledgments

During my Master’s studies I have received a lot of help from various persons. Most important by far have been the helps from my supervisors Professor Geoffrey Chan, and Professor Steve Blostein.

I would also like to thank all my current and past colleagues at Queen’s for all the valuable support both in a scientific as well as a social way. In particular I would like to thank: Javad for teaming up with me in almost every fun, Siavash for great parties. Alireza for teaching me basics of cooking, and conversations about goal of life. Mohsen for his support, and research discussions. Kamran for making fun of each other, cooking together, and parties. Carole for joining me in having evening coffee, and Omid for philosophical conversations.

I further wish to thank the faculty opponent and the committee for taking the time to be involved in the thesis defense.

Finally, I would like to thank my family back home: my dear mother, my dear father, and my dear little sister. They have supported me during these years from distance far away.
# Table of Contents

- **Abstract** i
- **Acknowledgments** ii
- **Table of Contents** iii
- **List of Figures** v

## Chapter 1: Introduction

<table>
<thead>
<tr>
<th>1.1 Source Coding</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2 Channel Coding</td>
<td>5</td>
</tr>
<tr>
<td>1.3 Binary Erasure Channel</td>
<td>8</td>
</tr>
<tr>
<td>1.4 Multiple Description Coding</td>
<td>9</td>
</tr>
<tr>
<td>1.5 Joint Source-Channel Coding &amp; MDC</td>
<td>26</td>
</tr>
<tr>
<td>1.6 Rate Distortion Region for Multiple Description Coding</td>
<td>30</td>
</tr>
<tr>
<td>1.7 Contributions</td>
<td>31</td>
</tr>
<tr>
<td>1.8 Organization of Thesis</td>
<td>32</td>
</tr>
</tbody>
</table>

## Chapter 2: Mixture Model-based Quantizers

<table>
<thead>
<tr>
<th>2.1 Gaussian Mixture Model</th>
<th>33</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2 Parametric vs. Non-Parametric Quantization</td>
<td>34</td>
</tr>
<tr>
<td>2.3 GMM Based Multiple Description Coding</td>
<td>40</td>
</tr>
<tr>
<td>2.4 Asymptotic Analysis of Multiple Description Coding Based on Mixture Models</td>
<td>45</td>
</tr>
<tr>
<td>2.5 Chapter Summary</td>
<td>58</td>
</tr>
</tbody>
</table>

## Chapter 3: G.711.1 Speech Coder

<table>
<thead>
<tr>
<th>3.1 G.711.1 Encoder</th>
<th>61</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.2 G.711.1 Decoder</td>
<td>67</td>
</tr>
</tbody>
</table>
## List of Figures

1.1 (a) Two independent uniform random variables before transform (b) Transformed random variables. .................................................. 11
1.2 MDSQ index assignment example (a) central encoder regions (b) side encoder regions (c) side encoder regions .................................. 16
1.3 Unequal loss protection .......................................................... 22
1.4 A toy FEC assignment example. ........................................... 23
1.5 Exponential PMF with a mean loss rate of 0.2 .......................... 24
1.6 Source utility vs. cost curve for Lenna image ............................ 25
1.7 ULP FEC assignment ......................................................... 26
1.8 ULP FEC assignment ......................................................... 27

2.1 Block diagram for cluster quantizer \( m \) ............................... 41
2.2 Mixture model-based multiple description coding system .......... 42
2.3 The effect of cluster probabilities on rate allocations .................. 51
2.4 Total average central and side distortion, and number of diagonals above the main diagonal in index assignment tables as functions of erasure probability ........................................ 56

3.1 Encoder block diagram ....................................................... 62
3.2 Lower-band embedded PCM encoder ................................... 63
3.3 Higher-band encoder ......................................................... 64
3.4 Sub-stream combination, and the corresponding overall bit rates for each supported mode of operation ......................... 67
3.5 Decoder block diagram ....................................................... 68

4.1 The effect of increasing \( \lambda \) on the MDSQ performance for a unit-variance memoryless Gaussian source ........................................ 73
4.2 MDSQ performance for ML(2,k) index assignment table ............ 74
4.3 MDSQ performance for ML(3,k) index assignment table ............ 75
4.4 MDSQ simulated average total distortion, and asymptotic average total distortion ...................................................... 76
4.5 Total average central and side distortions for three cases with different cluster separations ........................................ 77
Chapter 1

Introduction

Communication theory in its simplest terms is about a source of information communicating with a destination through a channel. A source can be any kind of signal like audio, speech, image, video, or data gathered from some sensors. A channel can be a wire, optic fiber, storage device, or air in a wireless communication. Channel specifications are not usually under the designer’s control. What a communication system designer can do with a given channel is to process source data to fit channel specifications, and at the same time, make them resilient against channel failures and errors. The former task is the topic of modulation and detection theory and the latter is the topic of coding theory.

Coding theory, as the name reveals, is about representing the source information in order to have reliable communication. Reliable communication is communication with arbitrarily small probability of symbol error. Obviously, the goal is to have such communication at an acceptable high rate. The possibility of having such reliable communication with rate greater than zero was under question until Shannon. Shannon founded information theory with which he showed that reliable communication
with positive rate is indeed possible.

1.1 Source Coding

In digital communication, two main processes are performed on source data before transmission through the channel. First, source data is converted into a binary sequence by a source encoder. Then the binary sequence is modulated for transmission over the channel. At the destination, the received signal from the channel is first demodulated to hopefully recover the original binary sequence, and the binary sequence is decoded by the source decoder to recreate the source output for the destination. The binary interface between source and channel encoder is standardized today for many reasons. Cheap and reliable digital hardware, independent source encoding from channel encoding and decoding, easy networking with binary sequences are among the reasons that make digital communication appealing today.

In source coding, the goal is to represent source information with a binary sequence with an average length as small as possible. In order to do so, source redundancy should be removed, i.e., the source data should be compressed as much as possible. To remove source redundancy, a quantitative measure of the average information that a source can provide is needed. This average information is called entropy of the source. For a discrete random variable $X$ with alphabet $\chi$, the entropy is defined as

$$H(X) = - \sum_{x \in \chi} p(x) \log_2 p(x).$$  \hfill (1.1)

The source coding theorem states that sequences of length $n$ from an independent and identically distributed (i.i.d.) process $X^n = (X_1, X_2, ..., X_n)$, drawn from probability distribution $p(x)$, can be represented using $nH(X)$ bits on average with
vanishing probability of error as \( n \) goes to infinity with a one to one and thus invertible mapping. The converse of the theorem proves that the source entropy is the infimum of the average code lengths, i.e., it is not possible to find a code with vanishing probability of error and average length smaller than the entropy.

For stationary random processes, it is possible to define a notion of entropy as the block length goes to infinity. This is called entropy rate, and the source coding theorem still holds for these sources.

### 1.1.1 Lossy Source Coding

Up to this point, we have assumed that sources are discrete with countable bounded alphabets. However, most of the time in practice sources are continuous, and can generate every possible value on an interval or the entire real line. Defining the true measure of information for these sources is not possible because the amount of information that they can provide is infinite. To make these sources tractable, we have to first quantize them into a discrete set of possible outcomes. By doing so, we are somehow reducing the entropy of the source; thus, the entropy that is defined for these sources is called differential entropy and this entropy can be negative.

It is obvious that by quantizing a continuous source into a discrete one, some of source information will be lost. Thus, quantization is a non-invertible mapping that introduces distortion. Many ways to quantify distortion are introduced in the literature depending on the application. Mean square error (MSE) is probably the most popular one, because it is more tractable and can lead to simple closed-form expressions. The MSE between a random variable \( X \) and its estimate \( \hat{X} \) is defined as

\[
D = E(X - \hat{X})^2. \tag{1.2}
\]
Quantization theory attempts to find ways to reduce the distortion for a given rate. The intriguing question of how much the distortion can be reduced for a given rate is the subject of rate distortion theory. For given rate and distortion measures, the rate distortion bound gives the least possible distortion that may or may not be achievable.

1.1.2 Scalar and Vector Quantization

The simplest way to quantize a continuous source is scalar quantization. The scalar quantizer of rate $R$ is a non-invertible mapping $Q$ from the real line, $\mathcal{R}$, onto the finite set of points $C = \{c_i\}$, $i \in \mathcal{I}$, where $\mathcal{I}$ is the set of indices with cardinality $2^R$, and $c_i \in \mathcal{R}$. $C$ is called the quantizer codebook. The quantizer divides the real line into regions (in the case of scalar quantizer intervals) $\mathcal{R}_i, i \in \mathcal{I}$. These regions are called quantization cells and are defined as: $\mathcal{R}_i = \{x \in \mathcal{R}, Q(x) = c_i\}$.

The goal in quantizer design with a given rate $R$ and distortion measure $d$ is to find the codebook $C$, and quantization regions $\mathcal{R}_i, i \in \mathcal{I}$ that minimize the average distortion between the quantizer input, random variable $X$ with density function $f_X(x)$, and quantizer output $\hat{X}$, the estimate of $X$. This optimization problem is very difficult and complex to solve in general. Instead, attempts have been focused more on finding locally optimal points rather than global ones. To this end, quantizer design is divided into two parts, namely, encoder and decoder design. Encoder design is dedicated to finding quantization cells for an already known codebook, and decoder design attempts to determine a codebook for known quantization cells. Sufficient conditions for optimality of the encoder and decoder have been derived for some well-known distortion measures, and by imposing those conditions back and forth on the
encoder and decoder designs, a locally optimum solution to the minimization problem can be found.

For the MSE distortion measure, with a fixed decoder, the nearest neighbor partition, and with a fixed encoder, the centroids of partition regions constitute necessary conditions for the optimality of encoder and decoder mappings, respectively.

Vector quantization is a generalization of scalar quantization. Instead of quantizing source samples one at the time as in scalar quantization, source samples can be concatenated to form vectors of dimensions more than one, and quantization can be performed on the resulting vectors. The design procedure is basically the same as for the scalar quantization encoder and decoder.

Vector quantization is the best that can be done for a source in terms of minimizing distortion for a fixed distortion measure. Less distortion is achievable by increasing the vector dimension at the cost of exponentially greater complexity. As more samples are included in one vector, source data dependencies between symbols, i.e., memory, may be exploited by the vector quantizer to lower distortion for a given rate. In addition, vector quantizers have the flexibility of cell shape over scalar quantizers. While cells are rectangular for multiple scalar quantizers on vector components, vector quantizers can employ non-rectangular cell shapes in higher dimensions to reduce distortion.

1.2 Channel Coding

A channel encoder is placed after the source encoder and before the channel modulator. In digital communication, a binary sequence that is generated by the source encoder is given to channel encoder as input.
The idea of channel coding is to add a controlled amount of redundancy to the symbols to provide resiliency against channel errors and/or failures at a cost of lowered communication rate. There are three main categories of channel codes: block codes, convolutional codes, and rateless codes. In this introductory section we only focus on block codes because the idea of a channel code can be best illustrated from block codes.

A block channel code of rate $R = \frac{k}{n}$, is an invertible mapping that maps sequences of length $k < n$ onto blocks of length $n$. $n$ is called the block length. Without loss of generality, we can assume systematic block codes for which each block of length $n$ consists of $k$ information symbols at the beginning of the block plus $n - k$ redundant symbols. The simplest block code of this kind is the single parity check code for which one redundant bit is added to each sequence of length $k$ in such a way as to make the number of ones in the whole sequence even or odd. This code has the ability to defeat any single error that has occurred in the sequence.

To have a measure of distance between two blocks of bits, one way is to count the number of positions that they differ. This measure is called Hamming distance between two blocks. For instance, the two sequences $\{010, 001\}$ differ in the second and third positions, so, the Hamming distance between them is 2.

It turns out that the correction and detection ability of a block code depends on its minimum Hamming distance. For a code with minimum Hamming distance $d$, the detection and correction capability are $d - 1$ and $t = \lfloor \frac{d-1}{2} \rfloor$, respectively. That is, a code is capable of correcting any error pattern with less than or equal to $t$ errors. Thus, in order to have more resilient blocks against channel errors, redundant symbols should be appended to the information symbols in such a way as to increase the
minimum Hamming distance of the resulting codebook.

One fundamental question in channel coding is: can we have reliable communication through a channel with a positive rate? By reliable communication, we mean arbitrarily small probability of error as the block length becomes larger. The question is answered by Shannon in the landmark paper [22]. Fortunately the answer is yes! The maximum amount of information that can be carried by a channel is computed as the supremum of the mutual information between its input and output over the input probability distribution, and is called the channel capacity. Suppose that $X$ is the input random variable with $p(x) = Pr\{X = x\}$ with alphabet $\chi$, and $Y$ is the channel output random variable with alphabet $\gamma$. The mutual information between $X$ and $Y$ is defined as

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X),$$

(1.3)

where $H(X|Y)$ is the conditional entropy of $X$ given $Y$ which is defined as

$$H(X|Y) = -\sum_{x\in\chi} \sum_{y\in\gamma} p(x, y) \log_2 p(x|y),$$

(1.4)

and $H(X)$ is the entropy of $X$ defined in (1.1). Channel capacity can be defined as

$$C = \sup_{p(x)} I(X;Y).$$

(1.5)

It can be proved that, for a discrete memoryless channel, all rates below channel capacity are achievable. That is, it is possible to find a block code with rate $R < C$, with arbitrarily small probability of error as the block length goes to infinity (Channel Coding Theorem).

Channel capacity is not easy to compute for every channel. Indeed, despite much effort, it has been only computed for a few types of channels. For instance, the capacity of relay channels in general form is still an open problem today. In this
thesis, packet erasure channels are assumed in the designs. So, let us elaborate on this channel and its capacity.

1.3 Binary Erasure Channel

The binary erasure channel (BEC) is a binary channel model. It can only transmit zero and one as its input, and the output is either the same as the input or erased. This erasure can happen for different reasons in practice depending on the application. For instance, in a network, erasure can be due to network congestion, buffer overflow, or in a wireless channel it may occur because of deep fades in signal power. The BEC can be viewed as a model of a physical process. For example, for a wireless system design one can declare signal power below some defined threshold as an erasure event.

The BEC channel is fully defined by erasure probability \( \alpha \). So, the probability for each input bit to be received without erasure at the output would be \( 1 - \alpha \). The capacity of the BEC is \([6]\)

\[
C_{\text{BEC}} = 1 - \alpha \quad \text{bits/channel use.}
\]  

(1.6)

Because the BEC is a binary channel, the best it can do is to send 1 bit for each channel use, and this would be achieved for zero erasure probability.

For packets of length \( L \) bits, if one or more bits are erased by the channel, packet erasure events will occur. For a memoryless BEC, the probability of such an event would be \( 1 - (1 - \alpha)^L \).
1.4 Multiple Description Coding

In multiple description coding (MDC) the idea is to represent a source with multiple descriptions in a way that enables recovery of the source information from each description with an acceptable amount of distortion. Moreover, we want our quality to improve smoothly as the number of received descriptions increases. These attributes are termed “good” descriptions. In layered coding the quality improves as more layers are received. However, unlike multiple description coding, later received layers would not be recoverable if former ones are erased or not received properly. In contrast, multiple descriptions (MDs) can be sent through independent channels to create diversity that can overcome channel impairments.

A MD coding system is difficult to design because of conflicting requirements. For instance, if one designs a good description with rate $R_1$, and another good description with rate $R_2$, there is no guarantee that these two descriptions, when received together, would be a good way to allocate total rate $R_1 + R_2$. On the other hand, it is not trivial to split a good representation at rate $R_1 + R_2$ into two useful descriptions.

Despite these difficulties, MD coding is a mature area, and a number of ways have been proposed in the literature for MD coding as well as its rate distortion region. In this section, we are going to explain the main MD coding schemes and associated rate distortion regions.

1.4.1 MD Coding Using Correlating Transforms

For a two description representation, the second description should add some extra information to the first one to improve its quality. At first glance, an independent description from the first one can add the maximum amount of extra information
to it. However, if two independent descriptions are too distinct, they would not both be good descriptions individually. This tradeoff is seen in all practical MD coding methods. One practical multiple description coding method is to use pairwise correlating transforms. We explain this method by the following example:

Suppose we have two independent random variables $x_1$ and $x_2$, with uniform distributions over the interval $[-1, 1]$. We use the correlating transform \[29\] to create the descriptions $y_1$ and $y_2$ in the following way:

\[
\begin{bmatrix}
  y_1 \\
  y_2
\end{bmatrix} = \begin{bmatrix}
  \theta (2\theta)^{-1} \\
  -\theta (2\theta)^{-1}
\end{bmatrix} \begin{bmatrix}
  x_1 \\
  x_2
\end{bmatrix},
\]

where $\theta$ is a positive real number.

Fig. 1.1(a) shows 1000 samples of $x_1$, and $x_2$ together in one graph, and Fig. 1.1(b) shows the transformed samples with $\theta = 4$. As can be seen from Fig. 1.1(a), knowing the value of $x_1$, does not give any information about $x_2$ as it can still take on any value from -1 to 1. However, in Fig. 1.1(b) knowledge of $y_1$, restricts the values of $y_2$ to a very short interval.

It can be shown that, $y_1$, and $y_2$ are correlated with correlation $E[y_1y_2] = -\theta^2\sigma_{x_1}^2 + (2\theta)^{-2}\sigma_{x_2}^2$, where $\sigma_{x_1}^2$ and $\sigma_{x_2}^2$ are the variances of $x_1$ and $x_2$ respectively. So, $\theta$ is the parameter that specifies the amount of correlation between the transformed random variables. On the other hand, the conditional expectation $E[x_2^2|x_1] = E[x_2^2] = 1/3$, and $E[y_2^2|y_1 = 0] = 1/3\theta^2$. That is, knowledge of $y_1$, reduces the standard deviation of $y_2$ as $\theta$ is increased. For instance, for $\theta = 4$, the variance of $y_2$ given $y_1 = 0$ is $1/16$ the variance of $x_2$. 
Now, let us interpret the above example from information theoretic view point. Because we are dealing with continuous random variables, by entropy we mean differential entropy, which can be negative.

The entropy of $x_1$, $x_2$, $y_1$, and $y_2$ would be:

$$h(x_1) = h(x_2) = 1$$

(1.8)

$$h(x_1, x_2) = h(y_1, y_2) = 2$$

(1.9)

$$h(y_1) = h(y_2) = 1 + \log_2 \theta + \frac{1}{4\theta^2}$$

(1.10)

all in bits per source sample.

From Equation (1.9), the joint entropy of the transformed descriptions is the same as the original ones, which is obvious because the correlating transform does not add new information. Rather, it just adds redundant information to the transform coefficients that makes these coefficients easier to estimate if either of them is lost. So, unlike the original $x_1$ and $x_2$ that were independent, the description $y_1$ does
CHAPTER 1. INTRODUCTION

contain information about the other description $y_2$. This can be seen by computing the mutual information between these random variables:

\[ I(y_1; x_1, x_2) = 1 + \log_2 \theta + \frac{1}{4\theta^2}, \quad (1.11) \]

\[ I(y_1; x_1) = \log_2 \theta + \frac{1}{4\theta^2}, \quad (1.12) \]

\[ I(y_1; y_2) = 2 \log_2 \theta + \frac{1}{2\theta^2}. \quad (1.13) \]

From Equations (1.11) and (1.12), it can be seen that $y_1$ has information about both $x_1$ and $x_2$, and from (1.13) $y_1$ has information about the other description $y_2$. This kind of information reflects the amount of redundancy in a multiple description representation. The amount of redundancy increases with $\theta$. Increasing $\theta$ makes descriptions better individually, but how about the case when we have both descriptions? From (1.9), it seems that having both descriptions gives us all the information we need regardless of the value of $\theta$.

However, we have not taken into account the effect of quantizing the transform coefficients. $y_1$ and $y_2$ are random variables with trapezoid shape pdf. They can take values from the interval $[-\frac{1}{2}(2\theta + \frac{1}{2}), +\frac{1}{2}(2\theta + \frac{1}{2})]$. Increasing $\theta$ would increase the length of this interval, and by using the same quantization rate we will have larger distortion for the case that we have both descriptions. This is the tradeoff that we mentioned at the beginning of this section. That is, although increasing $\theta$ makes better individual descriptions, because of the larger amount of redundancy amongst the descriptions, we end up with larger distortion for the case where we have both descriptions.
1.4.2 Multiple Description Scalar Quantizers

In [27], a multiple description scalar quantizer (MDSQ) is designed for the first time. In this section, we borrow notation and apply some of the results from this paper.

A diversity system with two independent channels is considered. The source is represented with two descriptions to transmit over two channels. If both descriptions are received without erasure, the output would be chosen from the central decoder. Otherwise, in case of one channel failure, the side decoder for the other channel would be used.

Specifically, let the diversity system consist of two encoders \( f_1: \mathcal{R} \rightarrow \mathcal{I}_1 \) and \( f_2: \mathcal{R} \rightarrow \mathcal{I}_2 \), one for each channel, where \( \mathcal{I}_1 = \{1, 2, \ldots, M_1\} \) and \( \mathcal{I}_2 = \{1, 2, \ldots, M_2\} \), and three decoders, two side decoders \( g_1: \mathcal{I}_1 \rightarrow \mathcal{R} \) and \( g_2: \mathcal{I}_2 \rightarrow \mathcal{R} \) for each channel output, and one central decoder \( g_0: \mathcal{C} \rightarrow \mathcal{R} \). \( \mathcal{C} \) is the central codebook, and a subset of \( \mathcal{I}_1 \times \mathcal{I}_2 \).

The system is fully identified by the central partition \( \mathbf{A} = \{A_{ij}, (i, j) \in \mathcal{C}\} \) on \( \mathcal{R} \), where \( A_{ij} = \{x : f_1(x) = i, f_2(x) = j\} \), an index assignment table that defines \( \mathcal{C} \), and decoder vector \( \mathbf{g} \).

The design objective is to find the encoder \( \mathbf{f} = (f_1, f_2) \), central partition \( \mathbf{A} \), codebook \( \mathcal{C} \), and decoder \( \mathbf{g} = (g_0, g_1, g_2) \) that minimize the average central distortion \( E(d_0) \) subject to the following constraints on average side distortions: \( E(d_1) \leq D_1 \) and \( E(d_2) \leq D_2 \). If random variable \( X \) represents the input to the MDSQ, and the output of decoder \( g_m, m = 0, 1, 2 \) is represented by the random variable \( \hat{X}^m, m = 0, 1, 2 \),
the Lagrangian functional for this optimization problem would be given by:

\[
L(A, g, \lambda_1, \lambda_2) = E \left(d_0 \left(X, \hat{X}^0\right)\right) + \lambda_1 \left(E \left(d_1 \left(X, \hat{X}^1\right)\right) - D_1\right) \\
+ \lambda_2 \left(E \left(d_2 \left(X, \hat{X}^2\right)\right) - D_2\right),
\]

where \(\lambda_1\) and \(\lambda_2\) are positive numbers.

The above optimization is tantamount to minimizing the total average distortion, when \(\lambda_1\) and \(\lambda_2\) are chosen in the following way [30]

\[
\lambda_1 = \lambda_2 = \frac{p}{1 - p},
\]

where \(p\) is the index erasure probability.

The above optimization, is difficult to tackle directly. In order to make the problem tractable, the optimal central partition (the optimal encoder) is found for fixed reconstruction points (decoder). Then for the resulting encoder, the optimal decoder is designed. Necessary conditions for encoder and decoder optimality is derived in [27].

Another issue to be determined is to find the optimal subset of \(I_1 \times I_2\), that is, to find an appropriate codebook, which is called index assignment optimization. In [27], two methods for index assignment are proposed heuristically, rather than finding an optimal index assignment.

Index assignment is basically defined by an index assignment matrix. For \(D\) descriptions, a \(D\)-dimensional matrix would be needed. Thus, for two descriptions with rates \(\log_2 M_1\) and \(\log_2 M_2\), the matrix has \(M_1\) columns and \(M_2\) rows. The main diagonal, and a number of diagonals parallel to it are good candidates for codewords. The number of diagonals is specified by parameter \(k\), which denotes the number of diagonals above the main diagonal. Notice that the main diagonal alone is for the most redundant case in which both descriptions are the same. In this case, we would have
the least side distortion, and the most central distortion. As the number of diagonals is increased, the codebook size increases, and less central distortion is achieved at the cost of increased side distortions. A simple index assignment example with equal side rates (for this example: $R_1 = R_2 = 2$) is shown in Fig. 1.2 as well as central and side partitions. The numbers in the index assignment table constitute the codebook for central encoder. For this example the side rate is $\log_2 4$. Numbers in each row and column correspond to side encoder for channel 1 and 2, respectively. For instance, to send index 8 to the destination, index 3 is sent through channel 1, and index 4 is sent through channel 2.

Index assignment optimization for multiple description vector quantizers is considered in [8]. An optimal decoder is designed for a given index assignment. Then, a quality criterion is derived for the optimization of index assignments. For $D$ descriptions, the optimal index assignment can be found by placing $|C|$ codewords into a $D$-dimensional matrix with $|I_1| \times |I_2| \times \ldots \times |I_D|$ locations such that the quality criterion is met. Instead of this brute force approach, the idea of a binary switching algorithm (BSA) is applied in the design to find a suboptimal index assignment matrix with less complexity.

### 1.4.3 Signal Processing-based MD Coding

MD coding based on signal processing is proposed in [13]. The proposed MD coding system is as follows:

The input is first decorrelated by a decorrelating transform like Karhunen Loeve Transform (KLT). Transformed coefficients are then decomposed into two subsources
by a polyphase transform. Each subsource forms the primary part of a single description. Each subsource is then quantized with two quantizers, $Q_1$ and $Q_2$, with rates $R$, and $\rho < R$, respectively. The output of the lower rate quantizer $Q_2$ for each subsource defines the redundancy that is multiplexed with the output of the quantizer $Q_1$ of the other one to create one description. Two descriptions are then sent through independent channels.

At the decoder, the received signal is demultiplexed into primary and redundant quantized samples. Samples are then decoded by their own dequantizers. Finally an inverse polyphase transform and KLT transforms are applied, respectively, to reconstruct data.

For the known primary rate $R$, there are two ways to find the redundancy rate $\rho$. One method is to minimize the central distortion with a constraint over the side distortion, and the other is to minimize the average total distortion. In the former,
the redundancy rate $\rho$ would depend on the upper limit of the side distortion, and for the latter it would depend on the channel erasure probability. Optimizations are performed under the high resolution quantization assumption.

The results are comparable to MDSQ results, with improved performance for very low primary and redundant bit rates.

### 1.4.4 Multiple Description Lattice Vector Quantization

Multiple description coding for scalar quantization can be generalized to be applicable for vector quantizers. The only modification in the formulations for vectors of length $N$ is to change the domain from real line to $N$ dimensional space $\mathbb{R}^N$.

If no constraints are added for partition structures, multiple description vector quantization (MDVQ) is in general not practical due to two reasons. First, the complexity grows exponentially as the vector dimension $N$ increases. Second, because there is no natural order in $\mathbb{R}^N$, design of index assignment matrices is more difficult.

Multiple description lattice vector quantization (MDLVQ) [21], [28] is a way to introduce structure to encoder design by exploiting the symmetries of lattices to avoid the above difficulties.

Lattices $\Lambda \subset \mathbb{R}^N$, and $\Lambda'$ are considered. $\Lambda'$ is a scaled and rotated sublattice of $\Lambda$. Indeed, $\Lambda$ is a fine lattice that defines the resolution for the central decoder, and $\Lambda'$ is a coarser lattice with the same structure that fixes the resolution for side decoders. For an MDLVQ system with $D$ descriptions, for each central decoder point of the lattice $\Lambda$, $D$ nearest neighbor points of the lattice $\Lambda'$ are considered as its descriptions.

Here, index assignment is a mapping from $\Lambda$ into $\Lambda' \times \Lambda'$, that can be defined on
an elementary cell, and extended to $\mathcal{R}^N$ due to lattice symmetries.

### 1.4.5 Multiple Description Coding Using Frames

Multiple description coding using frames is an MD coding technique with similarities to block channel coding [9], [10].

The idea is to left-multiply an $N$-dimensional source vector $X^N$ by the rectangular matrix $F \in \mathcal{R}^{M \times N}$, $M > N$ to create an $M$-dimensional descriptor vector $Y^M$:

$$Y^M = FX^N. \quad (1.16)$$

The resulting transform coefficients are scalar quantized to form descriptions. The data vector $X^N$ can be estimated from the quantized coefficients via a least-squares estimator

$$\hat{X} = \arg\min_X ||Y - FX||^2. \quad (1.17)$$

Loss of some of the descriptions in the channel is tantamount to erasing the corresponding rows from the rectangular matrix $F$ forming a new matrix $F'$. As long as the resulting matrix $F'$ has rank $N$, the reconstruction technique would not change. For the resulting matrix with rank less than $N$, the data vector can be estimated as:

$$\hat{X} = \arg\min_{X, F'X = Y} ||X||^2. \quad (1.18)$$

The similarity of this technique with block channel coding lies with the role of the frame operator $F$. The frame operator adds some redundancy to the $N$ symbols by introducing linear dependencies among $M$ transform coefficients. Unlike block codes that have discrete domain, here the domain is real valued.
1.4.6 Multiple Description Coding Using Channel Codes

The Internet is probably the most obvious example of a channel for which an erasure channel model is suited. To recover lost data or mitigate the loss impact for such erasure channels, existing techniques include automatic repeat request (ARQ) [12], forward error correction (FEC) [23] using error-correcting codes, and multiple description coding.

For FEC to provide sufficient protection with manageable delay and computational complexity for large bursts of bit erasures, block codes with block lengths $n$, and dimension $k$ need to have the capability to recover the received data from any subset of length $k$ of the $n$ encoded symbols.

Among linear block codes, this capability can be found in codes with maximum distance separation (MDS) property. For a linear block code $(n, k, d)$, we have:

$$d - 1 \leq n - k,$$

(1.19)

where $n$, $k$, and $d$ are the block length, code dimension, and minimum Hamming distance of the code, respectively. The bound in (1.19) is called the Singleton bound. MDS codes are linear block codes for which the Singleton bound holds with equality, that is,

$$d = n - k + 1.$$

(1.20)

It can simply be proven that, the only binary linear MDS codes are: $(n, n, 1)$, single parity code, and repetition code. Reed-Solomon (RS) codes are the well known class of non-binary MDS codes.

For RS codes, $n$ encoded symbols can be regarded as descriptors for $k$ information symbols. As long as the number of received symbols is greater than $k$, the receiver
can reconstruct the information symbols from them. If the number of received symbols drops below $k$, reconstruction quality drops dramatically, because the receiver is no longer capable of recovering the data except for the few symbols that might be received directly for systematic codes. This effect, namely the dramatic drop off in the reconstruction quality although many symbols might be received without erasure, is called the “cliff effect”.

The cliff effect is an unwanted situation that can happen in a download session as the destination cannot recognize the received stream although it has been receiving data for a significant amount of time because some packets are lost. The remedy for this problem is a technique in which the reconstruction quality (e.g. signal to distortion ratio (SDR)) improves smoothly as the number of descriptions is increased. This is the idea behind multiple description coding. The intriguing question here is: how is it possible to generate multiple descriptions by channel codes and to what extant would this method of doing multiple description coding be efficient? Can we use capacity achieving, strong channel codes to protect the data and, at the same time, benefit from multiple description coding advantages?

Multiple description coding using FEC has been studied in [1], [19], and [16]. In this thesis, we are going to compare a multiple description GMM quantizer that we have designed with the FEC based MD algorithm of [16] for the G.711.1 speech codec. We will refer to the algorithm proposed in [16] as “Mohr’s algorithm”, and in this section review some of the results from that paper.

The idea of Mohr’s algorithm is to exploit MDS error correcting codes, specifically Reed-Solomon codes, to construct multiple descriptions of the source data in such a way as to provide more protection for the more important source data. This
is done by adding more redundancy to the more important data by exploiting lower rate codes. In this way, the algorithm provides unequal loss protection (ULP) for the embedded input bit stream.

To this end, a layered encoded bit stream should be provided to the algorithm as input, because, for a layered bit stream, progressive layers have reduced levels of importance. The algorithm would know the importance of the input data from the contribution that it has on the overall reconstruction quality assuming that the more important symbols have already been recovered without error.

Fig. 1.3 depicts how ULP may be achieved by using RS codes as MDS codes. The most important part of the data stream is $M_1$ that needs the most loss protection. So, the most redundancy is added to this part of the data which is called $F_1$. The MDS property of the RS code implies progressive improvement as more packets are received. For instance, if up to 40% of the data is lost we would still be able to recover the most important part $M_1$, and if at least 80% of the data is received, we will not only be able to recover not only $M_1$ but also $M_2$. It can be seen in this figure that there are 3 different priorities assigned to $L$ streams. That is, $L_1$ of them are assigned the highest priority, $L_2 - L_1$ are assigned lower priority and so forth. Instead of this approach, greater priority can be assigned to symbol streams in such a way to maximize the expected peak signal to noise ratio (PSNR) [16] or other source quality criterion.

To see this more clearly, let us examine Fig. 1.4. In this toy example, 6 packets are to be transmitted each with 7 bytes. The goal is to maximize the expected PSNR of the received message as a function of $\bar{f}$ by summing over the $L$ streams:

$$G(\bar{f}) = \sum_{i=1}^{L} c(f_i)g_i(\bar{f}),$$  \hspace{1cm} (1.21)
where $\tilde{f} = (f_1, f_2, \ldots, f_L)$ is an L-dimensional FEC vector whose entries are the length of FEC assigned to each stream, $g_i(\tilde{f})$ is the amount of PSNR increase when the receiver decodes fragment $i$, given that all fragments prior to $i$ have already been decoded.

In Equation (1.21), the cumulative distribution function

$$c(f_i) = \sum_{n=0}^{f_i} p_n,$$

where $p_n$ is the probability that $n$ packets are lost. These probabilities are the estimated channel loss profile that a message is likely to encounter. The quantity $c(f_i)$ represents the probability that the receiver can decode stream $i$.

Mohr’s algorithm maximizes the expected PSNR (1.21) by a hill-climbing search technique under the constraint $f_i \geq f_{i+1}$ where $f_i$ is the number of FEC bytes assigned to stream $i$. To make the optimization problem tractable, at each iteration of the algorithm, $Q$ bytes of data can be added or subtracted to any of the $L$ streams.

The algorithm takes as its input: a probability mass function (PMF) of expected
channel/network loss conditions, a description of the source’s utility (PSNR/MSE) vs cost (bit rate, byte count, etc) curve and the length of each packet/description.

Fig. 1.5 shows an example of an exponential channel PMF with a mean loss rate of 20% as a function of the number of lost packets. The mean loss is computed as

\[ \frac{1}{N} \sum_{n=0}^{N} np_n = 0.2, \]  

(1.23)

where \( N \) is the total number of packets. The way the algorithm determines the PMF probabilities \( p_n \) is worth mentioning here. First it guesses an exponential PMF of the form:

\[ p_n = \exp(-\alpha n/N) / Z, \]  

(1.24)

where \( Z \) is the normalizing factor to make the probability sum to one and \( \alpha \) is a constant (the algorithm initializes \( \alpha = 10 \)). Then the algorithm verifies the guess by applying the initial PMFs to (1.23). If the result is more (less) than 0.2 with
some accuracy margin, the algorithm would increase (decrease) $\alpha$. With this search technique, it ends up with the profile of Fig. 1.5.

The other input is a description of the source’s utility (PSNR/MSE) vs cost (bit rate, byte count, etc) curve. A sample of this source utility is shown in Fig. 1.6 for a $512 \times 512$ gray scale Lenna image encoded by an SPHIT source codec. The numbers on the x-axis are normalized by 0.00003051758.

To understand Mohr’s algorithm better, let us observe how it works for an equal loss protection (ELP) scenario. For the ELP scenario, we are dealing with an $L$-dimensional FEC vector $\bar{f}$ whose entries are all the same, say $\beta$. The objective is to find $\beta$ that maximizes (1.21). Equation (1.21) for ELP can be re-written in the following form as a product of two terms:

$$G(\bar{f}) = c(\beta) \sum_{i=1}^{L} g_i(\bar{f}).$$

When $\beta$ is increased, i.e., when more protection is added to the streams, $c(\beta)$ would increase and the other term would decrease. A trade off exists between these
two terms. In some cases a global maximum can be achieved by Mohr’s search algorithm. Notice that, this rough explanation only holds when the source encoder provides a progressive bit stream, as in the case of a SPHIT encoder. Indeed, this “progressive” data is the main ingredient without which Mohr’s algorithm would be inapplicable. The optimization for the ELP scenario is performed by computer simulation on 137 packets, each with 47 bytes of data, for which $\beta$ was found to be 81. This means that, the algorithm would achieve a PSNR of 29.37 dB if the fraction of packets lost is less than 0.59. For packet losses of more than this amount, the ELP scenario performs the same as that of unprotected SPHIT.

Fig. 1.7 shows the ULP FEC assignment of Mohr’s algorithm for the above example. The progressive performance of the ULP algorithm can be seen in Fig. 1.8. As expected, the more packets received, the better quality can be achieved. Notice that the algorithm’s main objective is to find the FEC vector that maximizes (1.21). Actual protection of the data with RS codes or any other channel codes is not under
the control of the algorithm. Thus, the fact that Fig. 1.8 can be achieved with per-
fect protection, i.e., without any decoding failure or errors is idealistic. Another point
worth mentioning is that the ULP assignment of the Mohr’s algorithm maximizes the
“expected PSNR”. Indeed at some particular packet loss rates ELP performs better
than ULP.

1.5 Joint Source-Channel Coding & MDC

There are two common strategies for communicating analog data samples across a
binary channel, namely, tandem source-channel coding and joint source-channel cod-
ing.

In tandem coding a source code is designed without regard to the possibility of
channel errors. The objective for a source code design is to represent reliably the
source with as few bits as possible for every source sample. By reliable, we mean
that error probability becomes vanishingly small for infinitely large block lengths. In
tandem source-channel coding the source encoder is followed by a channel encoder in which a channel code is designed without regard to the nature of the source or the source code that produces the bits to be transmitted. The objective for a channel code design is to combat channel errors in the most efficient way possible. One way of doing this is to add redundancy to the bit stream. Channel codes combat channel errors at a cost of lowered communication rate.

Joint source-channel coding is an alternative strategy in which source and channel codes are jointly designed and/or operated to exploit the source statistics and to combat channel errors [15].

Shannon’s separation theorem [22] sheds light on the fact that no matter which of the two strategies is chosen, reliable communication is possible as long as the entropy of the source is smaller than the channel capacity. The separation theorem implies that, in the absence of delay and complexity constraints, no source-channel coding system could have better performance than the best tandem system! This fact is
somehow surprising because one might expect better performance for joint source-channel coding than tandem coding.

However, the separation theorem does not take into account constraints of real world communication systems. Delay and complexity are the two main constraints of any communication system that is intended to be used in practice. In [15], the performance, complexity, and delay of specific representative systems of tandem coding and joint source-channel coding strategies are quantitatively analyzed. The results suggest that joint coding yields less complexity and delay than tandem coding when and only when the target distortion is greater than a certain threshold, that depends on the source, channel, distortion measure, and type of source and channel codes considered. Complexity and delay are quantified, respectively, by the number of operations and the number of shift registers for each representation.

Results from [15] and similar other comparisons suggest that when complexity and delay are important constraints, we are better use joint source-channel coding approaches. Joint source-channel coding approaches may also give us a good perspective of system performance.

There are two main joint source-channel approaches that can be categorized as:

- Index assignment (IA)
- Channel optimized scalar/vector quantization (COSQ/COVQ)

In the index assignment approach, the encoder consists of a conventional quantizer followed by an IA which maps the index of each quantizer encoder partition region to multiple channel codewords. This method is basically multiple description scalar quantization (MDSQ). MDSQ can be implemented in two ways. First, the objective can be to minimize central distortion subject to an upper bound constraint for side
distortions to prevent them from exceeding a threshold. A second objective is to minimize total average distortion which depends on the channel index erasure probability. In the COSQ/COVQ approach [30], the channel condition is taken into account in the optimization. Thus, COSQ/COVQ can be regarded as a method of joint source-channel coding.

In the channel optimized approach, the encoder maps the source information directly to multiple channel codewords. Channel optimized scalar/vector quantization is a type of quantization for noisy channels. Let us assume $X$ to be the random variable that represents source output, $Y$ the quantizer output, and $\hat{X}$ the source data estimate at the output of the destination decoder. The mean-square error is given by (1.2), and can be decomposed into three terms [14]:

$$D = D_q + D_c + D_m,$$  \hspace{1cm} (1.26)

where $D_q$, $D_c$, and $D_m$ are, respectively quantization error, channel error, and mutual error, and are given by

$$D_q = E(X - Y)^2$$  \hspace{1cm} (1.27)

$$D_c = E(Y - \hat{X})^2$$  \hspace{1cm} (1.28)

$$D_m = 2E((X - Y)(Y - \hat{X})).$$  \hspace{1cm} (1.29)

The total distortion $D$ is a function of the probability density of $X$, $f_X(x)$, the channel transition matrix $P = \{P_{kl}\} = Pr[z = v_l|y = v_k]$ ($z$ is the channel output for input $y$), the quantizer decision levels $u_k$ ($k = 1, \ldots, M$), the quantizer values $v_l$ ($l = 1, \ldots, M$), and the number of quantization levels $M$. 

Taking the partial derivative of $D$, and equating it to zero the result is [14]:

$$v_j = \frac{\sum_{k=1}^{M} P_{kj} \int_{x_k}^{u_{k+1}} x f_X(x) dx}{\sum_{k=1}^{M} P_{kj} \int_{x_k}^{u_{k+1}} f_X(x) dx}, \quad j = 1, \ldots, M$$

(1.30)

$$u_j = \frac{1}{2} \sum_{l=1}^{M} v_l^2 (P_{jl} - P_{j-1,l})}{\sum_{l=1}^{M} v_l (P_{jl} - P_{j-1,l})}, \quad j = 1, \ldots, M.$$  

(1.31)

For a noiseless channel, (1.30), and (1.31) reduce to the centroid, and nearest neighbor partition conditions, respectively.

1.6 Rate Distortion Region for Multiple Description Coding

In [7], it is shown that for a sequence of i.i.d. random variables $X_1, X_2, \ldots, X_n$ with distortion measure $d(X_i, \hat{X}_i)$ on the estimates $\hat{X}_i$ of $X_i$, and two descriptions with rates $R_1$ and $R_2$ of the sequence $X = (X_1, X_2, \ldots, X_n)$, the distortion constraints $D_0, D_1, D_2$ are achievable if there exists a joint probability mass function $p(x, \hat{x}_1, \hat{x}_2, \hat{x}_0)$ with $E(d(X, \hat{X}_m)) \leq D_m, m = 0, 1, 2$ such that

$$R_1 > I(X; \hat{X}_1),$$

(1.32)

$$R_2 > I(X; \hat{X}_2),$$

(1.33)

$$R_1 + R_2 > I(X; \hat{X}_1, \hat{X}_2, \hat{X}_0) + I(\hat{X}_1; \hat{X}_2).$$

(1.34)

The rate-distortion region has been computed for a memoryless Gaussian source and squared-error distortion measure in [17]. It is shown that given $D_1 \geq 2^{-2R_1}$, and $D_2 \geq 2^{-2R_2}$, two descriptions with rates $R_1$ and $R_2$ exist, with average side distortions
CHAPTER 1. INTRODUCTION

no greater than $D_1$ and $D_2$:

$$E(d_1) \leq D_1,$$

$$E(d_2) \leq D_2,$$

and with average central distortion no greater than $D_0$, for any

$$D_0 > \begin{cases} \frac{2^{-2(R_1+R_2)}}{1-(\sqrt{\Pi}-\sqrt{\Delta})^2} & \text{if } \Pi \geq \Delta, \\ 2^{-2(R_1+R_2)} & \text{otherwise,} \end{cases}$$

where $\Delta = D_1D_2 - 2^{-2(R_1+R_2)}$ and $\Pi = (1-D_1)(1-D_2)$. Conversely, it is shown that given $D_1 \geq 2^{-2R_1}$ and $D_2 \geq 2^{-2R_2}$, it is infeasible to find two descriptions with rates $R_1$ and $R_2$, and with average side distortions that satisfy (1.35) and (1.36) and average central distortion smaller than the right hand-side of (1.37).

Again a tradeoff can be seen between side and central distortions. That is, for better average side distortions, i.e., smaller $D_1$ and $D_2$, a worse average central distortion would be obtained, and vice versa.

1.7 Contributions

This thesis devises quantization and source-channel coding schemes to increase the error robustness of the newly standardized ITU-T G.711.1 speech coder. The schemes employ Gaussian mixture model (GMM) based multiple description quantizers (MDQ).

Mixture model-based multiple description system is analyzed under a high rate assumption. Through asymptotic analysis of the general mixture model-based MD system, we propose rate and redundancy allocation formulations for both fixed-length and variable-length codes. We also propose algorithms to find integer numbers of bits
and amounts of redundancy that can be implementable in practical designs.

The effect of channel erasure probability on rate and redundancy allocations has also been investigated.

Two more widely used MD approaches namely FEC-based MD algorithm [16] and MMMDC [20] are evaluated and compared with one another on G.711.1 speech data as well as on model Gaussian data. In the thesis, for the FEC-based MD coding to be implementable on actual data, two layering approaches have been proposed to create an embedded bit-stream.

1.8 Organization of Thesis

Chapter 1 is an introduction to source and channel coding, multiple description coding, and its joint source-channel coding nature. Chapter 2 focuses on the review of GMM based quantizers and their applicability to the design of multiple description quantizers. Chapter 3 provides background on the G.711.1 speech coder. Chapter 4 focuses on the design and evaluation of GMM-MDQ schemes for G.711.1, and bit allocation for GMM-MDQ. GMM-MDQ with the proposed bit and redundancy allocations is compared with forward error correction (FEC) based MDQ in terms of G.711.1 error resilience improvement over packet erasure channels. Chapter 5 contains conclusions and summary of the results.
Chapter 2

Mixture Model-based Quantizers

2.1 Gaussian Mixture Model

A probability mixture model is a linear combination of probability distributions where all coefficients are non-negative and sum up to one. A linear combination with non-negative coefficients that sum up to one is called a convex combination.

Gaussian mixture model (GMM), as the name reveals, is a convex combination of Gaussian distributions. Gaussian distributions that constitute the sum are multi-variable distributions in general. For $D$-dimensional random vectors $\mathbf{X}$, a Gaussian mixture pdf is defined as:

$$f_{\mathbf{X}|\Phi}(\mathbf{x}|\Phi) = \sum_{i=1}^{M} \alpha_i f_{\mathbf{X}|\phi_i}(\mathbf{x}|\phi_i), \quad (2.1)$$

where $M$ is the number of components, and $\alpha_i$s are the component coefficients that are non-negative and sum up to one:

$$\sum_{i=1}^{M} \alpha_i = 1, \quad (2.2)$$
and the component densities \( f_{X|\phi_i}(x|\phi_i) \) are Gaussian:

\[
f_{X|\phi_i}(x|\phi_i) = f_{X|\mu_i,C_i}(x|\mu_i, C_i) = \frac{1}{(2\pi)^{D/2}|C_i|^{1/2}} e^{-\frac{1}{2}(x-\mu_i)^T C_i^{-1}(x-\mu_i)},
\]

with mean vectors \( \mu_i \) and covariance matrices \( C_i \). These components are sometimes referred to as clusters in the literature. A Gaussian mixture pdf can be completely defined by the following set:

\[
\Phi = \{M, \alpha_1, ..., \alpha_M, \phi_1, ..., \phi_M\} = \{M, \alpha_1, ..., \alpha_M, \mu_1, ..., \mu_M, C_1, ..., C_M\}.
\]

GMMs are among the most statistically mature methods for clustering, and they are also being intensively used for density estimation. Quantizers can be designed based on GMMs by exploiting their clustering or density estimation capabilities.

## 2.2 Parametric vs. Non-Parametric Quantization

There are basically two approaches to quantizer design, namely, parametric and non-parametric.

In the non-parametric approach, a training is employed for the design of the encoder and the decoder. Training data is a large database of samples representative of the source, chosen in a way to encompass the statistics of the source. After the training mode, there is a quantization mode in which the actual source data is encoded by the encoder and transmitted to the channel, and the decoder at the receiver recovers the source data. The encoder-decoder can remain in the quantization mode as long as the stationarity assumption for the source remains valid. Otherwise, the system should switch to the training mode again. Thus, the method is applicable if
the source statistics do not vary quickly in time. For instance, for fast fading wire-
less communication systems the non-parametric method might not be a good choice,
because the stationarity time of data is short.

Another problem in non-parametric design arises when the training set is small
relative to the size of the codebook. In this case, the training set is not large enough
to represent the source statistically, and as a result, the codebook tends to become
fitted to the training set only. Performance may be poor when the quantizer is eval-
uated on a disjoint set of source vectors [5].

If the probability density of the source is known, or can be estimated, a parametric
approach would be the remedy for the above problems. In this technique, the encoder
and decoder can be trained using samples that are drawn from a parametric proba-
bility density function instead of the training data, because, at least theoretically, it
would be possible to produce infinitely many training vectors. Here, the transmitter
only sends density parameters to the destination to train the decoder. This technique
is more bandwidth efficient than the non-parametric one, because, there are far fewer
density parameters than training data. For instance, for a source with Gaussian den-
sity, its mean and variance suffices for the destination to know the source density.

However, in practical applications, the density of the source can seldom be accurately
approximated by standard distributions. In these cases, mixture models like GMM
can be used to estimate the source probability density. It is proven that any contin-
uous pdf can be approximated arbitrarily closely by a Gaussian mixture density [24].

The problem of vector quantization based on GMMs is considered in [11]. There,
it is shown that high rate formulas can accurately predict the performance of model-
based quantizers, and a novel technique, an extension to conventional expectation
maximization (EM) is proposed for GMM parameter estimation. In the following section, we elaborate on some key results from [11].

2.2.1 Gaussian Mixture Based Quantizers

The expectation maximization (EM) algorithm is a widely used algorithm to estimate the parameter set \(\Phi\) that represents a Gaussian mixture pdf for a given number of components, \(M\). It is basically an iterative algorithm that aims to maximize the log-likelihood of the parameter set \(\Psi\):

\[
\Psi = \{\alpha_1, ..., \alpha_M, \phi_1, ..., \phi_M\}.
\] (2.5)

Each iteration of the algorithm guarantees an increase in the log-likelihood of \(\Psi\), that is, the likelihood of the set at iteration \(k + 1\) is greater than that of iteration \(k\) \((L(\Psi^{(k+1)}) \geq L(\Psi^{(k)})\)). For a given number of components \(M\), component coefficients, \(\alpha_i\), component means, \(\mu_i\), and component covariance matrices, \(C_i, i = 1, ..., M\) constitute the parameters that are to be estimated by EM algorithm.

The algorithm consists of two steps, namely, the expectation step and the maximization step. In the expectation step, the expectation of the log-likelihood of the complete data is taken using the GMM with parameters estimated from the previous iteration. The log-likelihood is a function of the GMM parameters. In the maximization step, the conditional mean from the expectation step is maximized with respect to the GMM parameters. The algorithm iterates between these two steps until convergence, and convergence to a local optimum is assured.
CHAPTER 2. MIXTURE MODEL-BASED QUANTIZERS

Given $N_x$ IID samples $\{x_n\}_{n=1}^{N_x}$, the EM algorithm for Gaussian mixtures iteratively yields estimates as follows [11]:

$$
\alpha_i^{(k+1)} = \frac{1}{N_x} \sum_{n=1}^{N_x} p_i^{(k)}(n),
$$

$$
\mu_i^{(k+1)} = \frac{\sum_{n=1}^{N_x} p_i^{(k)}(n)x_n}{\sum_{n=1}^{N_x} p_i^{(k)}(n)},
$$

$$
C_i^{(k+1)} = \frac{\sum_{n=1}^{N_x} p_i^{(k)}(n) (x_n - \mu_i^{(k+1)}) (x_n - \mu_i^{(k+1)})^T}{\sum_{n=1}^{N_x} p_i^{(k)}(n)},
$$

where probabilities $p_i^{(k)}(n)$ are defined as

$$
p_i^{(k)}(n) = \frac{\alpha_i^{(k)} f_{X|\mu_i^{(k)},C_i^{(k)}} (x_n|\mu_i^{(k)},C_i^{(k)})}{\sum_{j=1}^{M} \alpha_j^{(k)} f_{X|\mu_j^{(k)},C_j^{(k)}} (x_n|\mu_j^{(k)},C_j^{(k)})}. 
$$

A significant reduction in the number of parameters is obtained if the covariance matrix is assumed to be diagonal.

As discussed in [11], the shape of the mixture pdf only changes slightly for different initializations after many iterations. However, the individual parameter values may differ drastically.

An interesting comparison between parametric and non-parametric approaches appears in [11] through the quantization of line spectral frequencies (LSF) by 12-bit 5-D LSF quantizers. Two cases are considered for each approach, namely, the closed test and open test. In the closed test, quantization is performed over all the training data, while in open test, additional vectors other than training vectors are quantized.

It is shown that the performance of the parametric approach based on GMM is less dependent on the database size. For small numbers of training vectors, the closed test distortion of the GMM based quantizer is just slightly less than that of the open test, while this gap is larger between non-parametric open and closed test results.
Performance of the closed test in the non-parametric approach is much better than that of the parametric approach for small numbers of training vectors, while both the parametric closed and open tests outperform the non-parametric open test.

As the number of training vectors increases, the gap between closed and open tests becomes smaller for both quantization approaches. For large numbers of training vectors, the non-parametric approach outperforms the parametric GMM based approach for both closed and open tests.

Model-based quantizers can be decomposed into two blocks, namely, a density estimation block and a closed form quantization block. These two blocks can be optimized individually. The resulting modular system would be less complex, more bit rate scalable, and more adaptable for changes in source data statistics. However, performance is inferior to the performance of conventional quantizers.

The idea behind such block-based quantizers, first proposed in [25], is as follows: Training data is used for source density estimation in the density estimation block. The estimated density is a mixture Gaussian pdf with $M$ components or clusters. The resulting Gaussian mixture model can be represented as $M$ clusters with probability $\alpha_i$ for cluster $i, i = 1, ..., M$. The quantizer block can then be designed with the assumption that the encoder input is a D-dimensional Gaussian random vector generated from one of these $M$ clusters. For this assumption to be valid, the density estimation block should be able to estimate the mixture pdf with a large enough training data size, and the number of clusters $M$ should be chosen appropriately.

The quantizer block consists of $M$ cluster quantizers. It quantizes the received D-dimensional vector with all of the cluster quantizers individually, and picks the one with the least distortion. This way, the quantizer can find the specific cluster that
the received data comes from. In actual fact, the received data does not originate from only one cluster, but belongs to all clusters with different probabilities. The loose assumption of the data belonging to one cluster reduces system complexity at the cost of inferior performance compared to conventional quantizers.

Each cluster quantizer does the following: (i) subtracts the received data vector from the cluster mean $\mu_i$, $i = 1, ..., M$, (ii) decorrelates the centralized data using the KLT transform resulting in $D$ decorrelated transform coefficients, (iii) scalar quantizes the transform coefficients using $D$ scalar quantizers, (iv) correlates quantization coefficients using the inverse KLT transform, and (v) adds the cluster mean to the resulting vectors to create an approximate vector of the input data.

It has been show by Bennett in [4] that any scalar quantizer can be implemented by companding, and in [18], Panter and Dite found the compressor function that minimizes MSE. Thus, in each cluster, quantizer scalar quantization can be decomposed into a compressor and a uniform scalar quantizer. The quantized transformed coefficients should be decompressed by an expander before being given to the correlator as inputs. The remaining problem to be solved is the bit allocation among the $M$ clusters, and further among the $D$ uniform quantizers of each cluster. The bit allocation problem is treated in [25] for both fixed rate and variable rate cases.

If the scalar quantizers in the system of [25] are replaced with multiple description scalar quantizers (MDSQs), one would end up with a GMM based multiple description quantizer. The idea was first proposed in [20].
2.3 GMM Based Multiple Description Coding

Mixture model-based multiple description coding (MMMDC) is completely defined by a probability density estimator block that works based on GMM and models the source pdf with $M$ clusters. For $D$-dimensional source vectors, cluster $i, i = 1, ..., M$ has probability $\alpha_i$ and generates $D$-dimensional Gaussian random vectors each with mean $\mathbf{m}_i$ and covariance matrix $\mathbf{C}_i$. Thus, the system is completely defined by $M \times D$ multiple description scalar quantizers (MDSQs).

Fig. 2.1 shows the block diagram for one cluster quantizer, and Fig. 2.2 depicts the complete MMMDC quantizer. The input $D$-dimensional vector $\mathbf{x}$ is given to the MMMDC quantizer that comes from the density mixture estimator block. The input $\mathbf{x}$ is quantized with all $M$ cluster quantizers as shown in Fig. 2.2, and the one with the least total average distortion is chosen as the cluster of $\mathbf{x}$, and corresponding indices from that cluster are sent over channel 1 and channel 2 as two descriptions of that particular source vector.

Each cluster quantizer $m$ does the following as is shown in Fig. 2.1: the input vector is first subtracted from its mean $\mathbf{m}_m$, and the resulting vector is transformed via a linear decorrelating transform $Q_m^T$ (e.g. KLT). Transform coefficients $\mathbf{y}_m = (y_{m,1}, ..., y_{m,D}), m = 1, ..., M$ are then fed into $D$ multiple description scalar quantizers (MDSQs).

For each cluster $m$, the $d$th component $y_{m,d}, d = 1, ..., D$, is encoded by the MDSQ side encoders $i_1^{(m,d)} = E_1^{(m,d)}(y_{m,d})$ and $i_2^{(m,d)} = E_2^{(m,d)}(y_{m,d})$ operating at a side description rate $R^{(m,d)}$. The encoded indices are decoded by the side decoders $D_1^{(m,d)}, D_2^{(m,d)}$ and central decoder $D_0^{(m,d)}$ producing the estimates:

$$\hat{y}_1^{(m,d)} = D_1^{(m,d)}(i_1^{(m,d)}),$$

(2.10)
Figure 2.1: Block diagram for cluster quantizer $m$

$$y_2^{(m,d)} = D_2^{(m,d)}(i_2^{(m,d)}),$$  \hspace{1cm} (2.11)

$$y_0^{(m,d)} = D_0^{(m,d)}(i_1^{(m,d)}, i_2^{(m,d)}).$$  \hspace{1cm} (2.12)

These estimated transform coefficients are inverse transformed, and the mean $m_m$ is added to them to create corresponding source vector estimates

$$\hat{x}_1^{(m)} = Q_m(y_1^{(m)}) + m_m,$$  \hspace{1cm} (2.13)

$$\hat{x}_2^{(m)} = Q_m(y_2^{(m)}) + m_m,$$  \hspace{1cm} (2.14)

$$\hat{x}_0^{(m)} = Q_m(y_0^{(m)}) + m_m,$$  \hspace{1cm} (2.15)

where $\hat{y}_j^{(m)} = (y_j^{(m,1)}, \ldots, y_j^{(m,D)})$ \hspace{1cm} $j = 0, 1, 2.$

The total average distortion associated with each cluster is computed according to

$$d_m = (1 - p)^2 d(x, \hat{x}_0^{(m)}) + p(1 - p)d(x, \hat{x}_1^{(m)}) + p(1 - p)d(x, \hat{x}_2^{(m)}),$$  \hspace{1cm} (2.16)
where $p$ is the index erasure probability. Next, the cluster associated with input vector $x$ is found from

$$m^* = \text{argmin}_m (d_m).$$

(2.17)

The encoder sends indices $\{i_1^{(m^*,d)}\}$ and $\{i_2^{(m^*,d)}\}$ through channel 1 and channel 2, respectively. If the indices are sorted in ascending order, the decoder would be able to decode the received index pair by comparing them to the index boundaries associated with each cluster quantizer.

The remaining design issue that should be addressed is rate allocation among $M \times D$ MDSQs. In [20], rate allocation is proposed in the same way as in [25]. This is a two-step procedure. First, the optimal bit allocation scheme that minimizes the total average mean square distortion $D_{tot} = \sum_{m=1}^{M} \alpha_m D_m(b_m)$, subject to the fixed
rate constraint, \( 2^{b_{\text{tot}}} = \sum_{m=1}^{M} 2^{b_m} \) is given by:

\[
2^{b_m} = 2^{b_{\text{tot}}} \left( \frac{(\alpha_m c_m)^{D/D+2}}{\sum_{j=1}^{M} (\alpha_j c_j)^{D/D+2}} \right) \quad 1 \leq m \leq M, \tag{2.18}
\]

where \( D_m(b_m) \) is the high resolution expression in the D-dimensional Gaussian case,

\[
c_m = \left[ \frac{\prod_{d=1}^{D} \lambda_{m,d}}{c_m} \right]^{\frac{1}{D}}, \tag{2.19}
\]

where \( \lambda_{m,d} \quad m=1,\ldots,M, \quad d=1,\ldots,D, \) is the eigenvalue of the covariance matrix for dimension \( d \) in cluster \( m \).

The second step is to allocate \( b_m \) bits amongst cluster components. In [25] and [20], the KLT transform which is an optimal transform for Gaussian case is considered, and bit allocation is given as

\[
b_{m,d} = \frac{b_m}{D} + \frac{1}{2} \log_2 \left[ \frac{\lambda_{m,d}}{c_m} \right], \quad 1 \leq d \leq D. \tag{2.20}
\]

We note here that the KLT transform is the optimal transform with optimal bit allocation for multiple description coding as well as single description coding as is shown in [2].

Some points are not considered in the two-step rate allocation procedure of [20]:

1. According to the MDSQ design method proposed by Vaishampayan in [27], one should know the side rate, and the number of diagonals to be able to design an index assignment table. The latter is the parameter that specifies the amount of redundancy between two descriptions. In [20], the amount of redundancy for MDSQs index assignment tables (the number of diagonals for index assignment tables) is specified by training each MDSQ with all possible numbers of diagonals for a given channel erasure probability, and picking the one that minimizes total average distortion. The search over all possible numbers of diagonals for a given channel erasure probability makes the design procedure complex. We
show in the sequel that there is no need to train MDSQ for all possible numbers of diagonals for a given channel erasure probability. Alternatively, the number of diagonals for each MDSQ index assignment table can be specified as a function of channel erasure probability prior to training, and training can only be performed for that particular number of diagonal rather than all possible values.

2. There is no control over the amount of side distortion in this design. That is, the effect that the amount of side distortion might have on the rate and/or redundancy allocations is not known. For instance, if we want side distortion to be smaller than a predefined threshold, the design procedure in [20] does not provide us with optimal number of diagonals for index assignment tables that can minimize central distortion.

3. The optimal bit allocation is only considered for fixed rate quantizers. Variable rate quantization has remained an unsolved problem.

4. The rate allocation is only considered for Gaussian mixture models while it might be possible to derive a rate allocation formulation for general mixture models.

5. Rate allocation is only valid for the KLT, and is not applicable for general linear transforms.

6. The effect of cluster probabilities on rate and redundancy allocations is not studied in the design.

7. The effect of channel erasure probability is not studied in this design.
In the next section, we propose a design procedure for the MMMDC system that addresses the aforementioned problems. The design is based on high rate quantization theory.

2.4 Asymptotic Analysis of Multiple Description Coding Based on Mixture Models

For a balanced two-description scalar quantizer with side rate $R$, input random variable $x$ with variance $\sigma^2$, side estimates $\hat{x}_s, s = 1, 2$, and central estimate $\hat{x}_0$, it is shown in [2] that for the average central distortion $d_0 = E[(x - \hat{x}_0)^2]$ and the average side distortions $d_s = E[(x - \hat{x}_s)^2], s = 1, 2$ for any $a \in (0, 1]$, there exist coders for which at asymptotically high rates:

\[
\begin{align*}
    d_0 &= C\sigma^2 2^{-2R(1+a)}, & (2.21) \\
    d_1 &= S\sigma^2 2^{-2R(1-a)}, & (2.22)
\end{align*}
\]

where $C$ and $S$ are constants determined by source probability density function and encoding method. In [3], closed formulas for $S$ and $C$ are derived for general sources using asymptotic approximations for high rates. In (2.21) and (2.22), for $a = 1$, the central distortion $d_0$, has its smallest value when the side distortions are large. This is the no-redundancy situation in which the descriptions are independent. The parameter, $a$, specifies the amount of redundancy. Smaller values of $a$ correspond to higher redundancy levels since to achieve the same amount of central distortion, we need to make the rate $R$ larger. The number of diagonals in an index assignment table specifies the amount of redundancy. Let the number of diagonals be $2k + 1$. 
Then in order to have $a = 1/n, n > 1$, it is shown in [2] that it suffices to set

$$k = 2^{R/n}. \quad (2.23)$$

So, the smaller number of diagonals corresponds to the larger amount of redundancy. For instance, for balanced descriptions, having the main diagonal only would correspond to the highest level of redundancy because the second description is a copy of the first one. Equations (2.21) and (2.22) show that as $a$ increases (which means the redundancy decreases, the number of diagonals increases, and the dependency between descriptions decreases), side distortions become larger and the central distortion becomes smaller.

Now let us consider an MMMDC system with $M$ cluster quantizers. The cluster quantizer $m, m = 1, ..., M$ is shown in Fig. 2.1. A source sample vector $x = (x_1, x_2, ..., x_D)$, is then linearly transformed to the D-dimensional mean normalized vector $y_m$ through the linear transformation $Q^T_m$ (where $T$ denotes transpose):

$$y_m = Q^T_m(x - m_m). \quad (2.24)$$

For orthogonal transforms, $Q^T_m$, and MSE distortion, it is shown that distortions are equal in the original and transform domains [2]. Thus, from this point on we will assume the transforms $Q^T_m$ to be orthogonal. It is also assumed that the MDSQ for the $d$th transform coefficient of the $m$th cluster quantizer operates at a side rate of $R_{md}$.

For the MMMDC system, the average central and side distortions are respectively
CHAPTER 2. MIXTURE MODEL-BASED QUANTIZERS

given by

\[
D_0(R) = \sum_{m=1}^{M} \alpha_m d_0^{(m)}(R_m), \tag{2.25}
\]

\[
D_1(R) = \sum_{m=1}^{M} \alpha_m d_1^{(m)}(R_m), \tag{2.26}
\]

where \( R \) is an \( M \times D \) rate matrix with entries \( R_{md} \), \( R_m \) is the \( m \)th row of \( R \), and \( \alpha_m \) is the probability of the \( m \)th cluster. \( d_0^{(m)}(R_m) \) and \( d_1^{(m)}(R_m) \) are the average central and side distortions of the \( m \)th cluster, respectively, and are given by

\[
d_0^{(m)}(R_m) = \frac{1}{D} \sum_{d=1}^{D} d_0^{(d)}(R_{md}), \tag{2.27}
\]

\[
d_1^{(m)}(R_m) = \frac{1}{D} \sum_{d=1}^{D} d_1^{(d)}(R_{md}). \tag{2.28}
\]

From (2.21)-(2.23) and the fact that \( a = 1/n \) we have

\[
d_0^{(d)}(R_{md}) = C_{md} \sigma_{md}^2 2^{-2R_{md}(1+1/n_{md})}, \tag{2.29}
\]

\[
d_1^{(d)}(R_{md}) = S_{md} \sigma_{md}^2 2^{-2R_{md}(1-1/n_{md})} \tag{2.30}
\]

where \( n_{md} \) specifies the redundancy of the \( d \)th MDSQ of the \( m \)th cluster, \( C_{md} \) and \( S_{md} \) are constants that depend upon the pdf of the \( d \)th transform coefficient of the \( m \)th cluster.

After substituting (2.27)-(2.30) into (2.25) and (2.26), we end up with

\[
D_0(R) = \frac{1}{D} \sum_{m=1}^{M} \sum_{d=1}^{D} \alpha_m C_{md} \sigma_{md}^2 2^{-2R_{md}(1+1/n_{md})}, \tag{2.31}
\]

\[
D_1(R) = \frac{1}{D} \sum_{m=1}^{M} \sum_{d=1}^{D} \alpha_m S_{md} \sigma_{md}^2 2^{-2R_{md}(1-1/n_{md})}. \tag{2.32}
\]
2.4.1 Variable Rate Quantizer

In a variable rate quantizer, the average rate of the quantizer is fixed. The average rate for MMMDC system is defined as

\[ R_{av} = \frac{1}{D} \sum_{m=1}^{M} \sum_{d=1}^{D} \alpha_m R_{md}. \]  

(2.33)

Our goal is to find rate allocation matrix \( R \) which allows the minimization of the average central distortion \( D_0(R) \) subject to \( D_1(R) \leq d_1^* \) and \( R_{av} \leq R \).

We have the following Lagrangian:

\[ J = D_0(R) + \lambda_1(D_1(R) - d_1^*) + \lambda_2(R_{av} - R). \]  

(2.34)

As a necessary condition, taking the partial derivative of \( J \) with respect to \( 1/n_{md} \) and equating it to zero, we have

\[ \frac{1}{n_{md}} = -\frac{1}{4R_{md}} \log_2 \left( \frac{\lambda_1 S_{md}}{C_{md}} \right). \]  

(2.35)

After omitting \( \lambda_1 \) from the first condition, \( D_1(R) \leq d_1^* \), and (2.35), we have

\[ \frac{1}{n_{md}} = \frac{1}{2R_{md}} \log_2 \left( \frac{C_{md}^{1/2} D d_1^*}{S_{md}^{1/2} \sum_m \sum_d \alpha_m (C_{md} S_{md})^{1/2} \sigma_{md}^2 2^{-2R_{md}}} \right). \]  

(2.36)

If we substitute (2.36) into (2.23), we end up with

\[ k_{md} = \left( \frac{C_{md}^{1/2} D d_1^*}{S_{md}^{1/2} \sum_m \sum_d \alpha_m (C_{md} S_{md})^{1/2} \sigma_{md}^2 2^{-2R_{md}}} \right)^{1/2}, \]  

(2.37)

where \( k_{md} \) is the number of diagonals above the main diagonal in the index assignment table of the \( d \)th MDSQ of the \( m \)th cluster quantizer. From (2.37), if we assume the same type of quantizers for all \( M \times D \) MDSQs (e.g. level constraint), \( k_{md} \) is independent of \( m \) and \( d \). This means that for all MDSQs no matter what cluster quantizer they belong to the index assignment table has the same number of diagonals given by (2.37). Indeed, it is just the side rate (or the size of the table) that differs among different MDSQs.
If we substitute (2.36) into (2.34),
\[
J = \frac{1}{d^* D^2} \left[ \sum_m \sum_d \alpha_m (C_{md} S_{md})^{1/2} \sigma_{md}^2 2^{-2R_{md}} \right]^2 + \lambda_2 \left( \frac{1}{D} \sum_m \sum_d \alpha_m R_{md} - R \right).
\] (2.38)
By taking the partial derivative of \( J \) with respect to \( R_{md} \) and equating it to zero,
\[
R_{md} = \frac{1}{2} \log_2 \left[ \frac{2(C_{md} S_{md})^{1/2} \sigma_{md}^2 \ln 2}{\sqrt{d^* \lambda_2}} \right].
\] (2.39)
After substituting (2.39) into the second condition, \( R_{av} \leq R \), omitting \( \lambda_2 \), and using the fact that \( \sum_m \alpha_m = 1 \),
\[
R_{md} = R + \frac{1}{2} \log_2 \left[ \frac{\gamma_{md}}{(\prod_m \prod_d \gamma_{md})^{1/D}} \right],
\] (2.40)
where \( \gamma_{md} \) is defined as:
\[
\gamma_{md} = (C_{md} S_{md})^{1/2} \sigma_{md}^2.
\] (2.41)
Expression (2.40) enables us to compute the rate matrix \( R \) for an MMMDC system with a general mixture model. Note that \( R_{md} \) does not depend on the side distortion constraint \( d^*_1 \). This is also observed in [20], that the allocation is independent of the erasure probability.

If we assume all \( S_{md} \), and \( C_{md} \) for \( m = 1, ..., M \) and \( d = 1, ..., D \) to be constants \( S \) and \( C \) respectively, the minimum of \( D_0(R) \),
\[
D^*_0 = \frac{SC^{2^{-4R}}}{d^*_1} \left( \prod_m \prod_d \sigma_{md}^{2\alpha_m/D} \right)^2.
\] (2.42)
Thus, increasing side distortion, would increase the number of diagonals by the same amount in all MDSQs, and decrease the average central distortion of the system.

For a source with a Gaussian mixture pdf, and decorrelating linear transform (e.g. KLT) in each cluster quantizer, components in the transform domain become independent. All the constants \( S_{md} \) and \( C_{md} \) are equal and we have the following rate
CHAPTER 2. MIXTURE MODEL-BASED QUANTIZERS

allocation:

\[ R_{md} = R + \frac{1}{2} \log_2 \left( \frac{\sigma_{md}^2}{(\prod_m \prod_d \sigma_{md}^{2\alpha_m})^{1/D}} \right) \]  

(2.43)

Since, the geometric mean of the variances of the transform coefficients is minimized in each cluster quantizer for the KLT, KLT is optimal for this system for Gaussian mixture pdfs. That is, among all orthogonal transforms, for KLT \( R_{md} \) is maximum for all \( m \) and \( d \).

Now, let us investigate the effect of cluster probabilities on rate, side distortion, and central distortion for a GMM based MMMDC system with linear decorrelating transform for all cluster quantizers. To this end, suppose we have a Gaussian mixture pdf with 2 clusters with probabilities \( \alpha_1 \) and \( \alpha_2 \), each with one component. Further, we assume the variance of cluster 1 larger than that of the cluster 2 (\( \sigma_{11}^2 \geq \sigma_{21}^2 \)). We consider three cases:

1. \( \alpha_1 = 3/4 \quad \alpha_2 = 1/4 \):

\[
R_{11} = R + \frac{1}{4} \log_2 \frac{\sigma_{11}}{\sigma_{21}} \\
R_{12} = R + \frac{3}{4} \log_2 \frac{\sigma_{21}}{\sigma_{11}} \quad (2.44)
\]

2. \( \alpha_1 = \alpha_2 = 1/2 \):

\[
R_{11} = R + \frac{1}{2} \log_2 \frac{\sigma_{11}}{\sigma_{21}} \\
R_{12} = R + \frac{1}{2} \log_2 \frac{\sigma_{21}}{\sigma_{11}} \quad (2.45)
\]

3. \( \alpha_1 = 1/4 \quad \alpha_2 = 3/4 \):

\[
R_{11} = R + \frac{3}{4} \log_2 \frac{\sigma_{11}}{\sigma_{21}} \\
R_{12} = R + \frac{1}{4} \log_2 \frac{\sigma_{21}}{\sigma_{11}} \quad (2.46)
\]
Fig. 2.3 shows the rate allocation for these three cases. For all three cases $R$ and $d^*_i$ are the same values. The bar widths in the figure are proportional to the cluster probabilities and their heights are proportional to the cluster rates. As the probability of the second cluster with less variance ($\alpha_2$) increases, the rate for both clusters increases, and the redundancy in both of them decreases. Side distortions in both clusters decreases, but the average side distortion does not change. The average central distortion of the system decreases. We end up with a system with less average central distortion and the same average side distortion.
2.4.2 Fixed Rate Quantizers

For fixed rate quantizers, the total number of codepoints in the codebook is fixed. The total number of codepoints for an MMMDC system is

\[ 2^{R_{tot}} = \sum_{m=1}^{M} D \prod_{d=1}^{D} 2^{R_{md}}. \]  \hspace{1cm} (2.47)

The goal is to find rate allocation matrix \( \mathbf{R} \) that minimizes the average central distortion \( D_0(\mathbf{R}) \) subject to \( D_1(\mathbf{R}) \leq d_1^* \) and fixed rate constraint of (2.47). We have the following Lagrangian:

\[ J = D_0(\mathbf{R}) + \lambda_1 (D_1(\mathbf{R}) - d_1^*) + \lambda_2 \left( \sum_{m=1}^{M} D \prod_{d=1}^{D} 2^{R_{md}} - 2^{R_{tot}} \right). \]  \hspace{1cm} (2.48)

After taking the partial derivative of \( J \) with respect to \( 1/n_{md} \), equating it to zero, and omitting \( \lambda_1 \) from the first constraint, we end up with the same redundancy allocation as (2.36). Substituting (2.36) into (2.48) leads to:

\[ J = \frac{1}{d_1^* D^2} \left[ \sum_{m} \sum_{d} \alpha_m (C_{md} S_{md})^{1/2} \sigma_{md}^{2} 2^{-2R_{md}} \right]^2 + \lambda_2 \left( \sum_{m=1}^{M} D \prod_{d=1}^{D} 2^{R_{md}} - 2^{R_{tot}} \right). \]  \hspace{1cm} (2.49)

If we take the derivative of the Lagrangian with respect to \( R_{md} \) and equate it to zero,

\[ 2^{R_{md}} = \left( \frac{2 \alpha_m \gamma_{md}}{\lambda_2 \zeta_m D \sqrt{d_1^*}} \right)^{-\frac{1}{2}}, \]  \hspace{1cm} (2.50)

where \( \gamma_{md} \) is defined as in (2.41), and \( \zeta_m \) is defined as:

\[ \zeta_m = \prod_{d=1}^{D} 2^{R_{md}}. \]  \hspace{1cm} (2.51)

From (2.50)-(2.51),

\[ \zeta_m = \left( \frac{2 \alpha_m}{\lambda_2 D \sqrt{d_1^*}} \right)^{D-2} \eta_m, \]  \hspace{1cm} (2.52)

where \( \eta_m \) is defined as:

\[ \eta_m = \prod_{d=1}^{D} \gamma_{md}^{\frac{1}{D-2}}. \]  \hspace{1cm} (2.53)
From (2.47) and (2.52) we can compute:

$$\lambda_2 = \left[ \frac{2^R}{\sum_m \left( \frac{2 \alpha_m}{D \sqrt{d_1}} \right)^{\frac{D}{D+2}} \eta_m} \right]^{-\frac{D+2}{D}}. \tag{2.54}$$

After substituting \( \lambda_2 \) from (2.54) into (2.52),

$$\zeta_m = \frac{2^R(\alpha_m)^{\frac{D}{D+2}} \eta_m}{\sum_m (\alpha_m)^{\frac{D}{D+2}} \eta_m}. \tag{2.55}$$

Substituting (2.54)-(2.55) into (2.50), we end up with

$$R_{md} = \frac{R_{tot}}{D} + \frac{1}{2} \log_2 \frac{(\alpha_m)^{\frac{2}{D+2}} \eta_m}{\sum_m (\alpha_m)^{\frac{D}{D+2}} \eta_m} \frac{\gamma_{md}}{p}. \tag{2.56}$$

Expression (2.56) gives the rate allocation matrix for general mixture model based MMMDC systems with orthogonal linear transform for each cluster quantizer. For a Gaussian mixture based MMMDC system, and KLT transform for each cluster quantizer, the rate allocation matrix of (2.56) will give us the same result as when we substitute Equation (2.18) into (2.20).

Notice that redundancy allocation for the fixed rate quantizer is the same as that of variable rate quantizer. That is, for fixed rate quantizer, and the same quantizer types for all MDSQs (e.g. level constrained), the number of diagonals for index assignment tables is the same for all MDSQs.

### 2.4.3 Effect of Channel on System Performance

Let us consider an erasure channel with index erasure probability \( p \). The total average distortion for two balanced descriptions is given by:

$$D_{tot}(R) = (1 - p)^2 D_0(R) + 2p(1 - p) D_1(R) + p^2 \sigma^2, \tag{2.57}$$
where $\sigma^2$ is the source variance. The goal is to minimize $D_{tot}(R)$ with respect to both redundancy parameter $1/n_{md}$ and rate $R_{md}$ subject to the rate constraint $R_{av} \leq R$. $D_{tot}(R)$ from (2.57) can be rewritten in the following form:

$$D_{tot}(R) = (1 - p)^2 \left[ D_0(R) + \frac{2p}{1 - p} D_1(R) \right] + p^2 \sigma^2.$$ (2.58)

So, minimizing $D_{tot}(R)$ under the rate constraint is tantamount to minimizing the Lagrangian

$$J = D_0(R) + \rho D_1(R) + \lambda (R_{av} - R),$$ (2.59)

where $\rho$ is defined as

$$\rho = \frac{2p}{1 - p}.$$ (2.60)

If we take the derivative of $J$ with respect to $1/n_{md}$ and equate it to zero, we have

$$\frac{1}{n_{md}} = -\frac{1}{4R_{md}} \log_2 \left( \frac{\rho S_{md}}{C_{md}} \right).$$ (2.61)

Substituting (2.61) into (2.59),

$$J = \frac{2\sqrt{\rho}}{D} \sum_m \sum_d \alpha_m (C_{md} S_{md})^{1/2} \sigma_{md}^2 2^{-2R_{md}} + \lambda \left( \frac{1}{D} \sum_m \sum_d \alpha_m R_{md} - R \right).$$ (2.62)

Now, if we equate the derivative of $J$ with respect to $R_{md}$ to zero and omit $\lambda$ from the rate constraint, we end up with the same result as (2.40) for the rate allocation matrix. By substituting (2.40) and (2.61) into (2.58),(2.31),(2.32),

$$D_{tot} = 2\mu(p) 2^{-2R} \left( \prod_m \prod_d \gamma_{md}^{\alpha_m} \right)^{\frac{1}{\rho}} + p^2 \sigma^2,$$ (2.63)

$$D_1 = 2^{-2R} \frac{1}{\sqrt{\rho}} \left( \prod_m \prod_d \gamma_{md}^{\alpha_m} \right)^{\frac{1}{\rho}},$$ (2.64)

$$D_0 = 2^{-2R} \sqrt{\rho} \left( \prod_m \prod_d \gamma_{md}^{\alpha_m} \right)^{\frac{1}{\rho}}.$$ (2.65)
where \( \mu(p) \) is defined as
\[
\mu(p) = \sqrt{2p(1-p)}^3.
\] (2.66)

If we substitute (2.61) into (2.23), the number of diagonals above the main diagonal,
\[
k_{md} = \left( \frac{C_{md}(1-p)}{2pS_{md}} \right)^{\frac{1}{4}}.
\] (2.67)

For a zero-mean stationary Gaussian source with variance \( \sigma^2 \), and level-constrained (LC) MDSQs, all \( S_{md} \) and \( C_{md} \) terms are \( \pi \sqrt{27}/6 \) and \( \pi \sqrt{27}/24 \) [2], respectively. In this case, \( k_{md} = k \) for every \( m \) and \( d \) and
\[
k = \left( \frac{1-p}{8p} \right)^{\frac{1}{4}}.
\] (2.68)

From the above, the channel erasure probability does not have direct effect on the rate, and just affects the redundancy allocation. From (2.60), \( \rho \) is an increasing function of \( p \). Thus, increasing \( p \) increases the average central distortion \( D_0 \), and decreases the average side distortion, \( D_1 \). The average total central and side distortions are proportional to \( \mu(p) \) as equations (2.63)-(2.65) show. Fig. 2.4 plots \( \mu(p) \) and number of diagonals, \( k \) as functions of \( p \) on one graph. As \( k \) is a decreasing function of \( p \), as \( p \) increases, fewer diagonals are filled in index assignment tables, resulting in more redundancy, less side distortion, and more central distortion. \( \mu(p) \) is an increasing function before its maximum. Its maximum occurs at \( p = 0.25 \), and after the maximum it is a decreasing function of \( p \). So, for \( p \leq 0.25 \), the effect of the average central distortion on the total distortion is dominant, and for \( p > 0.25 \) the effect of the average side distortion is dominant on the average total distortion.

\( k \) represents the number of diagonals above the main diagonal in an index assignment table. However, expression (2.68) does not necessarily give us an integer value for \( k \). So, the remaining problem that should be taken care of here is: how to
find an optimal integer value for $k$? For a case like zero-mean stationary Gaussian source with variance $\sigma^2$ and level-constrained MDSQs, $k_{md} = k$ is independent of $m$ and $d$. If we put $k$ from (2.23) into (2.57), for optimum value of $R_{md} = R^*_md$ we have

$$D = (1 - p)^2 \frac{A}{k^2} + 2p(1 - p)Bk^2 + p^2\sigma^2,$$

(2.69)

where $A$ and $B$ are given by

$$A = \frac{C}{D} \sum_{m=1}^{M} \sum_{d=1}^{D} \alpha_m \sigma^2_{md} 2^{-2R^*_md},$$

(2.70)

$$A = \frac{S}{D} \sum_{m=1}^{M} \sum_{d=1}^{D} \alpha_m \sigma^2_{md} 2^{-2R^*_md},$$

(2.71)

with the assumption that constants $S_{md}$ and $C_{md}$ are all equal to $S$ and $C$, respectively. We further assume that rounding $k$ to the nearest integer does not significantly change the total average distortion of (2.69). For any fixed $p$, the function $D$ is a convex function of $k$ with a minimum in the value of $k$ given by (2.68). The second order
derivative of $D$ with respect to $k$ is

$$\vartheta(k) = \frac{\partial^2 D}{\partial k^2} = 6(1 - p)^2 \frac{A}{k^4} + 4p(1 - p)B. \quad (2.72)$$

$\vartheta(k)$ is always positive, and a decreasing function of $k$. That is, the value of the function in $k^+ = \lceil k \rceil$ is smaller than the value of the function in $k^- = \lfloor k \rfloor$, where $\lceil \rceil$ and $\lfloor \rfloor$ are the ceil and floor functions, respectively. This means that, rounding $k$ to $k^+$ increases the total average distortion of (2.69) much less than $k^-$. Thus, $k$ should always be rounded to the smallest integer greater than or equal to $k$, and $k_{\text{int}}$ is given by

$$k_{\text{int}} = k^+ = \lceil k \rceil. \quad (2.73)$$

A similar problem occurs with the rate allocation matrix. The rates $R_{md}$ from (2.40) are not necessarily integers, and we need to assign integer numbers of bits for each MDSQ. Equation (2.40) is derived from taking the partial derivative of (2.62) with respect to $R_{md}$ and equating it to zero. In creating the Lagrangian $J$ of (2.62), the goal was to optimize the following function under rate constraint $R_{av} \leq R$:

$$Q(R) = \frac{2\sqrt{p}}{D} \sum_m \sum_d \alpha_m \gamma_{md} 2^{-2R_{md}}. \quad (2.74)$$

This function is monotonically decreasing with respect to $R_{md}$ for every $m$ and $d$. Thus, a good way to obtain integer rates is to round the optimal $R_{md}$'s from Equation (2.40) to the smallest integer greater than $R_{md}$ for every $m$ and $d$. However, the rate constraint $R_{av} \leq R$ will not hold any more, and $R_{av}$ will become greater than $R$. To solve this problem, we subtract one bit from those $R_{md}$'s that have the least contribution in increasing $Q(R)$, and re-check the rate constraint. If we take the derivative of $Q(R)$ with respect to $R_{md}$ at the optimal rate points of (2.40), $R_{md}^*$, we
have
\[
\frac{\partial Q(R)}{\partial R_{md}} \bigg|_{R_{md}=R^{\star}_{md}} = \frac{-4ln2}{D} \sqrt{\rho} \left( 2^{-2R_{\alpha_{m}}} \left( \prod_{m} \prod_{d} \gamma_{md}^{\alpha_{m}} \right) \right)^{\frac{1}{2}}.
\] (2.75)

The slope of the function \( Q(R) \) at the optimal rate points only depends on the probability of the cluster of those points. So, clusters with lower probabilities can be the first candidates for rate reduction. Thus, we propose the following algorithm to find integer optimal rates:

1. Find optimal non-integer rates from Equation (2.40).
2. Round all of the rates to the smallest integers greater than the optimal rates.
3. Arrange cluster probabilities \( \alpha_{m}, m = 1, ..., M \) in ascending order, and for each cluster \( m \), arrange the corresponding variances \( \sigma_{mi}^{2}, i = 1, ..., D \) in ascending order.
4. Start from the least probable cluster and a component with the least variance in that cluster, and reduce the corresponding rate by one bit. Do not reduce rates more than once.
5. Check the average rate \( R_{av} \). If it is less than \( R \) then stop, else go to Step 4.

With this algorithm, we would be able to find optimal integer rates for all MDSQs.

2.5 Chapter Summary

The MMMDC system is consisted of \( M \times D \) MDSQs. If one can allocate rate and redundancy among the MDSQs in such a way as to minimize the MMMDC total average distortion, MMMDC design would be simplified to designing individual MDSQs,
a topic addressed in [27].

Rate allocation problem is addressed in [20] under the assumption of decorrelating transform, and only for Gaussian mixture model (GMM). In [20], redundancy allocation among MDSQs, i.e., finding the number of diagonals for MDSQs index assignment tables, is performed through searching among all possibilities, and picking the one that minimizes total average distortion. This search makes the design procedure complex.

In this chapter we analyzed the performance of the MMMDC system [20] under high rate assumption. Through the asymptotic analysis, we gain new insight into the performance of the MMMDC. Asymptotic analysis enables us to perform rate allocation for MMMDC system with general mixture model, and general orthogonal transform for both fixed rate and variable rate constraints. The analysis proves that the number of diagonals for MDSQs index assignment tables should be the same for all MDSQs, and the value depends on channel erasure probability. Rather than searching among all possibilities to find the optimal number of diagonals for index assignment tables, we find the number from (2.67), and the algorithm that we proposed to round the number to an integer.
Chapter 3

G.711.1 Speech Coder

In Chapter 2, we derived the design equations for a mixture model-based multiple description coding system. In this chapter, we will review the G.711.1, a recent speech coding standard, that has been approved by ITU-T in March 2008 [26]. Our goal is to improve the codec’s performance over packet erasure channels through the use of a mixture model-based multiple description coding system.

G.711.1 is the extension to ITU-T G.711. It is an scalable coder that has been standardized for wideband telephony and voice over IP applications. Its extended feature is to give wideband scalability to G.711, that makes it applicable for broadband voice over IP applications.

Multiple description coding for G.711.1 data enables us to design a system that puts the advantages of multiple description coding, and the applicability of G.711.1 for broadband voice over IP together. That is, a speech codec can be designed which is (i) resilient against packet erasures, (ii) capable of being used in multicast applications with smooth improvement in its performance (e.g. SDR) as more packets are received, (iii) applicable for broadband voice over IP applications.
In this chapter, we borrow notation from [26], the ITU-T standard manual for G.711.

3.1 G.711 Encoder

The high-level encoder block diagram is shown in Fig. 3.1. The input signal to the encoder can either be a high-band or narrow-band signal. To describe all the encoder features, we can assume a high-band input signal. The 16-kHz-sampled input signal is filtered by a high-pass pre-processing filter to remove its 0-50 Hz components. The filtered signal is given to a 32-tap quadrature mirror filterbank (QMF), and is divided into 8-kHz-sampled lower-band and higher-band signals. The two resulting lower-band and higher-band signals are treated separately using lower-band and higher-band processing cores, respectively. The lower-band encoder is compatible with the G.711 core bitstream. The lower-band encoder generates two layers, namely, layer 0 ($I_{L0}$) at 64 kbit/s, and lower-band enhancement bitstream, layer 1 ($I_{L1}$) at 16 kbits/s. The higher-band encoder on the other hand, generates the higher-band bitstream $I_{L2}$. All of these layers $I_{L0}$, $I_{L1}$, and $I_{L2}$ are then multiplexed to generate a scalable bitstream.

3.1.1 Lower-band Encoder

The lower-band signal from the output of the filterbank is encoded using an embedded μ-law or A-law pulse coded modulation (PCM). Noise feedback is added to the PCM encoder that perceptually shapes its coding noise. The output of the noise shaping
filter is added to the lower-band signal before quantization. Fig. 3.2 shows the lower-band embedded PCM encoder block diagram. As it is shown in the figure, noise shaping is done using a linear prediction (LP) filter that is applied to the quantization noise. The filter impulse response in the $z$-transform domain is given by

$$F(z) = \sum_{j=1}^{L} \gamma^j a_j z^{-j},$$

where $\gamma$, $L$, and $a_j$ are the weighting factor, filter order, and filter coefficients, respectively. Filter coefficients $a_j$ are computed from the quantized signal $\hat{s}_{L0}(n)$ using the perceptual filter calculation block in order to have the same filter in the decoder as in the encoder. $L = 4$ is the typical filter order that is used, and 0.92 is a typical value for the weighting factor $\gamma$.

Extension bits are added to layer 0 to create layer 1 in order to enhance the
quality of the lower-band encoded bitstream. Adding layer 1 to layer 0 increases the number of steps in the quantizer and leads to finer quantization at the cost of 80 more bits for each 5 ms frame. For an 8-kHz-sampled lower-band signal, each 5 ms frame consists of 40 samples, and each sample is encoded with 8 bits in layer 0, and 2 bits in layer 1 that leads to 320 bits in total for layer 0, and 80 bits for the enhancement layer 1. The 8 bits for layer 0 include 1 bit to represent the sign, 3 exponent bits that indicate a compander segment, and 4 bits to indicate a given position within the compander segment.

### 3.1.2 Higher-band Encoder

The higher-band signal is transformed into the modified discrete cosine transform (MDCT) domain by the MDCT block. Transform coefficients $S_{HB}(k), k = 0, ..., 39,$ are then encoded using the higher-band MDCT encoder to generate the higher-band bitstream $I_{L2}$.
The input to the MDCT block is the 8-kHz-sampled higher-band signal $s_{HB}(n)$. The transform operation is done on frames of length 5 ms, and an analysis window of length 10 ms. Each frame consists of 40 samples, and the analysis window length is 80 samples. Coefficients are computed with 50% block overlap. MDCT transform coefficients are computed as:

$$S_{HB}(k) = \frac{1}{80} \sum_{n=0}^{79} w_{TDAC}(n) \cos \left( \frac{\pi}{40} (n + 20.5)(k + 0.5) \right) s_{HB}(n) \quad k = 0, ..., 39 \quad (3.2)$$

where $w_{TDAC}$ is the analysis window given by:

$$w_{TDAC}(n) = \sqrt{2} \sin \left( \frac{\pi}{80} (n + 0.5) \right) \quad n = 0, ..., 79. \quad (3.3)$$

The transform coefficients are quantized using interleave conjugate-structure vector quantization (CSVQ) in the higher-band encoder block. The block diagram of the higher-band encoder is shown in Fig. 3.3.

The MDCT coefficients are first weighted in the frequency domain to generate
weighted coefficients $S_{HBw}(k)$ in the following way:

$$S_{HBw}(k) = \begin{cases} 
0 & 0 \leq k \leq 3, \\
w_{HB}(5) \sin \left( \frac{\pi}{24} (2(k - 4) + 1) \right) S_{HB}(k) & 4 \leq k \leq 9, \\
w_{HB}(4) S_{HB}(k) & 10 \leq k \leq 15, \\
w_{HB}(3) S_{HB}(k) & 16 \leq k \leq 21, \\
w_{HB}(2) S_{HB}(k) & 22 \leq k \leq 27, \\
w_{HB}(1) S_{HB}(k) & 28 \leq k \leq 33, \\
w_{HB}(0) S_{HB}(k) & 34 \leq k \leq 39 
\end{cases} \quad (3.4)$$

where the frequency window $w_{HB}(j)$ is defined as follows:

$$w_{HB}(j) = \begin{cases} 
1 & j = 0, \\
1 & j = 1, \\
4/3 & j = 2, \\
5/3 & j = 3, \\
2 & j = 4, \\
1 & j = 5. 
\end{cases} \quad (3.5)$$

The RMS calculation block calculates the RMS gain as follows:

$$g_{HB} = \sqrt{\frac{1}{36} \sum_{k=0}^{35} (S_{HBw}(k + 4))^2}. \quad (3.6)$$

MDCT coefficients are then normalized by the RMS normalization factor, $g_{HB}$, as:

$$\bar{S}_{HBw}(k) = \frac{S_{HBw}(k)}{g_{HB} + \epsilon}, \quad (3.7)$$

where $\epsilon$ is a small number to avoid zero division. The gain $g_{HB}$ is quantized by the 8-bit gain quantizer to generate $I_{HG}$, because, the decoder should know the gain in
order to obtain $S_{HBw}(k)$ from $\tilde{S}_{HBw}(k)$.

The normalized transform coefficients $\tilde{S}_{HB}(k)$, are quantized by the interleaved CSVQ encoder in the following way: the normalized transform coefficients are first decimated to create six sub-vectors with dimension six ($S'_{HB}(v, j) \quad v = 0, ..., 5, j = 0, ..., 5$). Each sub-vector is quantized by using a two-channel CSVQ. Code vectors are computed by taking the average of two sub-code vectors with polarities $P_{H0}(v)$ and $P_{H1}(v)$ that can either be 1 or -1. The code indices $I_{Hs0}(v)$ and $I_{Hs1}(v)$, and the polarities are selected in way to minimize the following distortion:

$$d_{HB}(I_{Hs0}(v),I_{Hs1}(v), P_{H0}(v), P_{H1}(v)) = \sum_{j=0}^{5} \left( S'_{HB}(v, j) - \frac{P_{H0}(v)C_{H0w}(I_{Hs0}(v), j) + P_{H1}(v)C_{H1w}(I_{Hs1}(v), j)}{2} \right)^2,$$

where $C_{H0w}(I_{Hs0}(v))$ is the code vector with index $I_{Hs0}(v)$ in sub-codebook 0, and $C_{H1w}(I_{Hs1}(v))$ is the code vector with index $I_{Hs1}(v)$ in sub-codebook 1.

To reduce system complexity, the selection process is divided into two steps, pre-selection, and main selection. In pre-selection, 8 candidates are chosen among 32 codevectors in each codebook channel in such a way to minimize the Euclidean distance between target sub-vector and code vector. After pre-selection, the best pair that can minimize the distortion in expression (3.8) are chosen among all feasible combinations of selected code vectors.

In each sub-vector, for MDCT vector quantizer indexes, $I_{Hs0}(v)$ and $I_{Hs1}(v)$, 5 bits are dedicated, and for MDCT vector signs, 1 bit is allocated. This leads to a total of 80 bits for layer 2.

There are 4 modes of operation, namely, $R_1$, $R_2a$, $R_2b$, and $R_3$ depending on
which layers are involved in the embedded bitstream. Notice that in all modes, layer 0 is involved as it is the base layer, and the two other layers, layer 1 and layer 2, are enhancement layers. Fig. 3.4 shows the sub-stream combination as well as the corresponding overall bit rate for each mode.

### 3.2 G.711.1 Decoder

The high-level decoder block diagram is shown in Fig. 3.5. A multiplexed bitstream is given to the decoder, and is de-multiplexed into three bitstreams, namely, the core bitstream $I_{L0}$, the lower-band enhancement bitstream $I_{L1}$, and the higher-band bitstream $I_{L2}$. The two layers associated with the lower-band encoder are decoded by the lower-band decoder core, and the higher-band bitstream is decoded by the
3.2.1 Lower-band Decoder

The lower-band decoder basically does the reverse operation of the lower-band encoder. The block that should be described here is the lower-band frame erasure concealment (FERC) block. When a frame associated with lower-band signal is erased due to channel failure, this block reconstructs the lower-band signal. First, linear
predictive coding coefficients, pitch, and signal class for the missing frame are estimated from analyzing the past lower-band signal. There are four signal classes: voiced, weakly voiced, unvoiced, and transient. Then, the missing frame is synthesized from the estimated parameters of the previous step using LPC-based pitch repetition and adaptive muting [26]. If a good frame is received, the synthesized frame is re-synchronized with the new properly decoded signal.

3.2.2 Higher-band Decoder

Higher-band decoder does the reverse operation of the higher-band encoder. We now describe FERC block.

When a frame associated with a higher-band signal is erased due to channel failure, higher-band FERC reconstructs the higher-band signal. Two methods for reconstruction are considered by the block depending on the higher-band signal exhibiting high or low correlation.

For low correlation higher-band signals, the signal is considered as a noise-like higher band signal. In this case, an attenuated inverse MDCT signal from the last good frame is copied and used as the inverse MDCT data for erased frames.

In case of high correlation higher-band signals, a higher-band pitch lag is estimated around the lower-band pitch lag. The lower-band pitch lag is computed using lower-band FERC for the lower-band signal and given to the higher-band FERC. Samples of the previous pitch period from the last properly decoded higher-band frame in higher-band FERC history buffer are used as the basic inverse MDCT data of the current erased frame.
3.3 G.711.1 & Multiple Description Coding

In Chapter 2, we analyzed the MMMDC system performance under high rate assumption. Through our analysis, we proposed a design procedure to design the MD-based system with lower complexity.

In Chapter 3, we investigate G.711.1 speech coder in more detail. G.711.1 is an extension to G.711 with an extended feature of wideband scalability that makes it applicable for broadband voice over IP applications. To be able to use G.711.1 for packet erasure channels (e.g., Internet), G.711.1 wideband speech data should be made resilient against packet losses. Ideally, the quality in terms of SDR should be improved gradually as more packets are received. This is important for a multicast scenario specially when the number of users is large. Multiple description coding is suited for this purpose.

To protect G.711.1 speech data by MD coding, we propose to put MD coder in place of higher-band encoder of G.711.1 which is based on conjugate structure vector quantizer (CSVQ). This way, higher-band data can be protected by MD coding, and higher-band MDCT coefficients would be the data for MD coder to be protected. In Chapter 4, FEC-based MD coding, and mixture model-based MD coding are MD coders considered for G.711.1 data protection. First, the two MD approaches are compared for an example of a source pdf with two Gaussian components. Second, the two MD approaches are compared for an example of G.711.1 higher-band data.
Chapter 4

Simulation Results

Our goal is to enhance the error resilience of G.711.1 speech coder with multiple description coding. Two multiple description coding approaches are considered: multiple description coding based on forward error correction (FEC) codes as in Mohr’s algorithm, and mixture model-based multiple description coding, i.e., the system that was designed in Chapter 2. Our objective is to determine which multiple description coding scenario can best enhance the performance of G.711.1 speech coder over packet erasure channels, and in what sense this enhancement can be achieved.

Both multiple description approaches are based on the fact that the source pdf is estimated a priori as a mixture model Gaussian pdf through the expectation maximum (EM) algorithm.

4.1 Multiple Description Scalar Quantizers

In this section, we simulate the performance of multiple description scalar quantizers, which form the building blocks of the MMMDC system.
As explained in Section 1.4.2, the goal in the MDSQ design is to find the encoder and decoder pair in such a way to minimize central distortion under constraints on side distortions. That is, to minimize the Lagrangian in Equation (1.14). In the following we assume $\lambda_1$ and $\lambda_2$ to be equal to the constant $\lambda$, and investigate the effect of the parameter $\lambda$ on the system performance.

For a given rate and redundancy allocation (the number of diagonals, above the main diagonal), the MDSQ obtained with $\lambda = 0$ is a Lloyd-Max quantizer, because, MDSQ design in this case does not take into account side distortion and it just minimizes central distortion. MDSQ is trained to minimize the Lagrangian in Equation (1.14) for each value of $\lambda$, and $k = 1$. As $\lambda$ is increased, central distortion increases and side distortion decreases. Fig. 4.1 shows the effect of increasing $\lambda$ on MDSQ performance for a unit-variance memoryless Gaussian source with side rate of $R = 1$ bps. The index assignment is modified linear ML(1,1) (the first parameter in parentheses is side rate and the second one is $k$) [27].

To see the effect of $k$ on system performance and check the validity of our MDSQ design, let us consider ML(2,k) as our index assignment family. Now, the side rate is $R = 2$ bps and we can have two values $k = 1, 2$. For $k = 1$, we have the most redundancy and thus expect to have the least side distortion. As $k$ increases, the number of index pairs in the codebook increases and the redundancy in the index assignment table decreases. So, we expect to have less central distortion and more side distortion as $k$ increases from 1 to 2. Fig. 4.2 shows MDSQ performance for unit-variance memoryless Gaussian source for both values of $k$ [27]. Rate distortion region of (1.37) is also plotted in the figure (OPTA). Fig. 4.3 shows MDSQ performance for a unit-variance, memoryless Gaussian source with index assignment ML(3,k), for
Figure 4.1: The effect of increasing $\lambda$ on the MDSQ performance for a unit-variance memoryless Gaussian source.

$k = 1, 2, ..., 6$ \cite{27}.

4.2 Data Model

To gain a better understanding of the performance of multiple description algorithms, let us start with a source with a Gaussian mixture pdf consisting of two components each with dimension 2. From now on, let us name each Gaussian component a cluster. So, we have two clusters. The probability of the first cluster is 0.75 and the probability of the second one is 0.25. This means that from 10000 2-D source vectors, approximately 7500 are from the first cluster and 2500 are from the second one.
The covariance matrix of the first cluster

\[ C_1 = \begin{bmatrix} 256/3 & 0 \\ 0 & 16/3 \end{bmatrix} \]  \hspace{1cm} (4.1)

and for the second one

\[ C_2 = \begin{bmatrix} 16 & 0 \\ 0 & 1 \end{bmatrix} \]. \hspace{1cm} (4.2)

From the above covariance matrices it is obvious that the two components in each cluster are assumed to be independent.

### 4.2.1 Cluster Bit Allocation

We want to allocate bits to cluster components in a way so as to minimize the total average distortion of (2.57) under the average rate constraint \( R_{av} \leq R \). The allocation matrix \( R \) can be computed from (2.43) for the variable length case.

For this example, let us consider \( R = 3 \), and find integer rates from (2.43) using...
CHAPTER 4. SIMULATION RESULTS

Figure 4.3: MDSQ performance for ML(3,k) index assignment table

the algorithm proposed in Section 2.4.3. We end up with the following integer rate matrix

\[
R = \begin{bmatrix}
4 & 2 \\
3 & 1 
\end{bmatrix},
\]

(4.3)

We also determine the number of diagonals for the index assignment tables from the algorithm proposed in Section 2.4.3. Fig. 4.4 shows the total average distortion as a function of index erasure probability. The asymptotic curve of (2.63) is also shown in the same graph. Here, the cluster means are assumed to be \( \mu_1 = [0 \ 0] \) and \( \mu_2 = [3 \ 4] \). The quantizers in this example are level constrained quantizers for which the constants \( S \) and \( C \) are \( \pi \sqrt{27}/6 \) and \( \pi \sqrt{27}/24 \), respectively. The figure shows that the asymptotic curve of equation (2.63) can closely predict the distortion of the MMMDC system. The approximation becomes more accurate for larger erasure probabilities.

To see the effect of multi-modality of the source pdf on system performance,
we can change the cluster means. Fig. 4.5 shows the total average central and side distortions as functions of index erasure probability for three cases. For case I: $\mu_1 = [0 \ 0]$ and $\mu_2 = [3/2 \ 2]$, for case II: $\mu_1 = [0 \ 0]$ and $\mu_2 = [3 \ 4]$, and for case III: $\mu_1 = [0 \ 0]$ and $\mu_2 = [6 \ 8]$. Thus, for case III, the two clusters are separated the most from each other. The shape of the graphs in all cases is the same as predicted by the first term in Equation (2.63) which is proportional to the function $\mu(p)$ in (2.66). It is shown in the figure, that as the clusters come closer together, distortion reduces. That is, for a fixed erasure probability, the least total average central and side distortion can be achieved for case I in which the two clusters are closest to each other compared to the other two cases. For two clusters that are far apart from each other, the size of boundary cells between the two for each cluster is bigger, and these large boundary cells are responsible for larger distortion. Notice that the discontinuity in the figure is due to averaging over small number of data points. The point that
we investigate in this figure is the effect of cluster separation (multi-modality of the pdf) on the total average central and side distortions.

4.3 Fixed length MDSQ vs. FEC-based MD coding for GMM

In this section we use the simple data model of Section 4.2 to compare the performance of mixture model-based MD coding with that of the FEC-based MD coder over the packet erasure channel. For the mixture model-based MD coding system we use fixed length codes, and follow the formulations derived in Section 2.4.2 to allocate rates and redundancies to the two components of the two clusters.

For this comparison, let us assume system side rate of $R_{s,tot} = 4$. The optimal
matrix $\mathbf{R}$ from (2.56) is

$$
\mathbf{R} = \begin{bmatrix}
2.8390 & 0.8390 \\
1.8390 & -0.1610
\end{bmatrix}
.$$  

(4.4)

In Section 2.4.3, we proposed an algorithm to find optimal integer bit allocations for variable rate constraint. However, in this section we are dealing with fixed rate constraint. To find integer rates, we have to round some of the rates to the upper integer and some to the lower one, and check the rate constraint to be sure that the number of codewords are less than or equal to $2^{R_{\text{stot}}} = 16$ (fixed rate constraint: $2^{R_{\text{stot}}} \leq 16$). By following the same procedure as in Section 2.4.3 for variable rate constraint, to find integer rates for fixed rate constraint, we take the derivative of the first term in (2.49) with respect to $R_{md}$ in the optimal rate points of (2.56). We see that for the cluster with larger $\alpha_{m}^{D_{m}} + 2_{m}^{\eta_{m}}$, the distortion change in the optimal point is greater. For our example, this cluster is the first cluster. Thus, we round all the rates to the upper integer, and by starting from the rates of the second cluster reduce component rates by one and check the number of codewords. We stop rate reductions when the fixed rate constraint is fulfilled, i.e., we reach a number of codewords below 16. Notice that we are allowed to reduce each rate by no more than one bit. By this procedure, we end up with the following integer rate matrix:

$$
\mathbf{R}_{\text{int}} = \begin{bmatrix}
2 & 1 \\
1 & 0
\end{bmatrix}
.$$  

(4.5)

We assume here that each packet consists of 47 bytes or $47 \times 8 = 376$ bits. Each index or source vector can be represented using 4 bits, so, there are two indices in each byte and a packet can handle 94 indices. Thus, we should parse the side distortion every 94 indices. If we parse the central distortion every 94 indices, we end up with an
updated central distortion value for every 2 received packets. To packetize encoded
data, the first received packet is side 1 of the first 94 vectors, the second received
packet is side 2 of the first 94 vectors, and the third received packet is side 1 of the
second 94 vectors, and so forth. Thus, after receiving four consecutive packets the
decoder would be able to decode 184 vectors with the best quality, because it has
received both sides of those 184 vectors without erasures.

The decoder is initialized with 2000 zero vectors. Each received index can update
one of the 2000 decoder vectors. To be able to compare this algorithm with FEC-based
MDC algorithms, we want to have some distortion measure which is monotonically
increasing like signal to distortion ratio (SDR) as a function of the number of received
packets. We can put the SDR in dB as

\[
SDR = 10 \log \left( \frac{\alpha_1 \text{max} \{\sigma_1^{(1)}, \sigma_1^{(2)}\} + \alpha_2 \text{max} \{\sigma_2^{(1)}, \sigma_2^{(2)}\}}{\text{MSE}} \right),
\]

(4.6)

where \(\sigma_j^m\) is the standard deviation of the \(j^{th}\) component of the \(m^{th}\) cluster.

Fig. 4.6 depicts the central SDR in dB as a function of the number of received
packets.

### 4.4 FEC-Based Multiple Description Coding with Multi-Stage Layering

To gain insight into how Mohr’s algorithm searches for the optimum point to maximize
the expected peak signal to noise ratio (PSNR) of the received message, we provide
an example in this section. PSNR is generally defined as the ratio between maximum
signal power and noise power.
Figure 4.6: MDSQ central SDR in dB as a function of number of received packets

The expected PSNR is given as [16]:

$$G(\bar{\mathbf{f}}) = \sum_{i=1}^{L} c(f_i)g_i(\bar{\mathbf{f}}),$$  \hspace{1cm} (4.7)

where $L$ is the packet length in bytes, $c(f_i)$ is the probability that the receiver can decode stream $i$ and $g_i(\bar{\mathbf{f}})$ is the amount by which the PSNR increases when the receiver decodes fragment $i$, given that all fragments prior to $i$ have already been decoded [16]. The expected PSNR is a multivariate function of

$$\bar{\mathbf{f}} = (f_1, f_2, \ldots, f_L).$$  \hspace{1cm} (4.8)

In (4.8), $f_i$ is the number of FEC bytes assigned to stream $i$. To be able to conveniently plot the PSNR, we assume only two streams of length $N$ and by changing the FEC allocations between the two, we plot the PSNR as a function of number of allocated data bytes for each stream. Fig. 4.7 shows such a curve. In this figure, the z-axis is the expected PSNR in dB for which we are looking for an optimum point, and the x-axis and y-axis represent the numbers of data bytes allocated to the first
Figure 4.7: 3D contour of the Lenna SPIHT with two streams.  

and second streams, respectively. Exponential channel loss profile 20% mean loss is used in the simulation. The data is Lenna encoded with set partitioning hierarchical trees (SPIHT) which is a wavelet-based image coding algorithm. Fig. 4.8 shows the top view of Fig. 4.7. In Fig. 4.7, the expected PSNR is plotted for different data allocations two streams, and only five contours of a resulted surface is plotted in the figure. The algorithm iterations are shown in Fig. 4.8 as well. It can be shown that the algorithm path is perpendicular to the contour curves. This means that the algorithm changes in the gradient direction. The convergence point at the bottom of the figure is the final point in which the algorithm converges.

Our aim is to generate multiple descriptions of the source using FEC codes. One approach to do this, is by using the Mohr’s algorithm. Basically, the Mohr’s algorithm receives a layered coded bit stream as an input and generates multiple descriptions from that by adding FEC bytes to each stream at the cost of rate loss. The significance of the algorithm lies on the maximum distance separation (MDS)
property of the channel code with which each stream is coded. Reed Solomon codes are the well known codes that have the MDS property.

We now propose a layered coding scheme that can be used as an input to the Mohr’s algorithm. By a layered coded stream we mean that each layer is decodable only when all previous layers have successfully been received. The more layers successfully received, the higher the decoding quality. The encoder should inform the decoder of the cluster number to which each source vector belongs. This task is done in the first layer. The first layer is basically an entropy coder. From now on let us assume that $x_i$ and $y_i$ are, respectively, the first and second components of cluster $i$.

Equation (4.5) gives side rate allocations for index assignment-based MD coding system. For fixed length codes, total number of bits for the first cluster and second cluster are 6 and 2, respectively. This leads to 68 codewords for index assignment-based MD system. To have a fair comparison between FEC-based MD coding, and index assignment-based MD coding, the number of codewords must be the same for them. From Equation (2.56), and the algorithm that gives integer rates for fixed length codes, we have the following integer rate matrix for FEC-based MD coding.
system:

\[
R_{FEC} = \begin{bmatrix}
3 & 2 \\
2 & 1 \\
\end{bmatrix}.
\]  

There are four symbols to be coded in the first layer namely:

1. The left half of cluster 1 for which \(x_1 \leq 0\), \(p_1 = 0.75 \times 0.5 = 0.375\)

2. The right half of cluster 1 for which \(x_1 > 0\), \(p_2 = 0.75 \times 0.5 = 0.375\)

3. The left half of cluster 2 for which \(x_2 \leq 0\), \(p_3 = 0.25 \times 0.5 = 0.125\)

4. The right half of cluster 2 for which \(x_2 > 0\), \(p_4 = 0.25 \times 0.5 = 0.125\)

where \(p_i\) represents the probability for the \(i^{th}\) symbol.

The entropy of this source is 1.8113 bits per source symbol and the average length of the first layer is 1.8670. Notice that after decoding the first layer, the decoder would know two pieces of information about each source vector: (1) to which cluster it belongs to and (2) to which half of that cluster it belongs to. For example, if the decoder decodes symbol 3 for the first source vector it will know that the first vector belongs to the left half of the second cluster. Now the encoder starts encoding the second layer. We have 2 bits left for \(x_1\), 2 bits for \(y_1\), 1 bit for \(x_2\) and 1 bit for \(y_2\). For every source vector, this layer sends 1 bit for each component. The encoder moves the cartesian origin to a point on the x-axis in such a way that the number of quantization regions in all four cartesian regions be equal. Notice that because of layer 1, the encoder knows whether the origin should be moved to the right half or the left. For example, suppose that the first source vector is in the right half of the first cluster. In that case, the origin should be moved in the positive direction. Suppose further that the step size for the x-component is \(\Delta_x\). If we move the origin to \(\Delta_x\),
we end up with 6 quantization regions for the first cartesian region, 2 regions for the second, 2 for the third and 6 for the fourth so this is not a good choice! If we move the origin to $+2\Delta_x$, instead, we end up with 4 quantization regions in each cartesian region so this is a good choice. Indeed, this illustrates a rule for moving the cartesian origin for each layer. As long as the region is specified with the previous layer, the next one can move the origin according to this rule. Now that the encoder has moved the cartesian origin, it should check the cartesian region in which the source vector lies. For the first region $(x > 0, y > 0)$ it will send 01, for the second $(x < 0, y > 0)$ 00, for the third $(x < 0, y < 0)$ 10, and for the fourth $(x > 0, y < 0)$ 11.

By the end of the second layer, 1 bit is left for $x_1$ and 1 bit for $y_1$ and no bits are left for the second cluster. Thus, the third layer is dedicated to the first cluster only. For the third layer we proceed as in layer 2. Namely, we move the cartesian origin to a point in such a way that the number of quantization regions in all cartesian regions is the same and then send two bits for the cartesian region to which the source vector belongs.

Fig. 4.9 shows the SDR in dB for the layered bit stream as a function of the number of received bytes. Indeed, this is the profile that can be used as an input to the Mohr’s algorithm. Notice that the break in the curve occurs at the end of the first layer.
4.5 FEC-Based Multiple Description Coding with Bit-Plane Layering

Mohr’s algorithm gets a layered bit stream as its input, and generates multiple descriptions of a source using FEC. In Section 4.4, we proposed multi-stage layering as an approach to generate a layered bit-stream from source data. In this section, we are going to propose bit-plane layering as another way to generate an embedded bit-stream. Our goal is to investigate the effect of layering method for Mohr’s algorithm input on the resulting multiple description SDR performance. Particularly, by comparing two proposed layering approaches, we investigate the property that lies in SDR performance of the input to the Mohr’s algorithm that makes FEC-based multiple description SDR curve more smooth as a function of the number of received packets, i.e., more resilient against “cliff effect”.

For the first cluster, we have a total of 5 bits and for the second one, we have a total of 3 bits. So, there are 32 indices in the first cluster and 8 indices in the second one.
For the first layer the encoder just specifies the cluster number. It sends zero for the first cluster and one for the second one. So, after the first layer the decoder would know the cluster number.

We enumerate the quantization regions for the first cluster as follows: We start from top left quantization region and call it zero. Then we come to the right of that region and add the number until we reach the y-axis. Then we go to the second row. After we finish the second cartesian region, we start with the top left region of the first cartesian region and proceed analogously. Then we go to the third and fourth regions respectively. At the end, we will have 0-7 in the second region, 8-15 in the first, 16-23 in the third and 24-31 in the fourth. We enumerate the second cluster analogously and end up with 0-1 in the second region, 2-3 in the first, 4-5 in the third and 6-7 in the fourth. Fig. 4.10 shows cluster enumeration for both clusters. The figure only depicts enumeration, and quantization regions are not necessarily the same as in the figure.

Now, the encoder starts scanning from the most significant bit of each source...
vector index and sends it in the second layer. So, after decoding the second layer, the decoder can distinguish between 0-15 and 16-31, i.e., it finds out the sign of the y-axis. After three subsequent layers no bits would left for the second cluster and there are still 2 bits left for the first cluster for which two more layers are needed.

Fig. 4.11 shows the SDR of this layered bit stream as a function of number of received bytes. Again, the first break point is after layer 1 is received. Notice that unlike the previous multi-stage layering approach, we cannot see any improvement in the system performance through the first layer. This is because in this case, layer 1 has no information other than cluster means, and cannot update the decoder from its zero condition (in this case, the clusters are zero mean). Layer 1 can only inform the decoder of the cluster number to which a particular source vector belongs.

Let us compare the three aforementioned algorithms namely bit-plane layered, multi-stage layered, and MDC without FEC. The channel model used in this comparison is the erasure channel with uniform packet loss with the mean loss rate of 20%. For the layered schemes the profiles of Figures 4.9 and 4.11 are given to the Mohr’s algorithm as inputs to produce multiple descriptions of the source.
Fig. 4.12 shows all three algorithms in one plot as functions of number of received packets. The MDC curve is the average distortion with $p$ the probability of packet erasure. The multi-stage layering algorithm outperforms the bit-plane one and this is because the bit-plane algorithm should receive many of its layers to yield the same resolution as the multi-stage.

Another point to notice here is that with this number of received packets, the two FEC based algorithms are unable to recover the whole data and this is why the end points for these two algorithms are smaller than that of the MDC without FEC. After a few more packets, the two FEC based curves eventually jump to the point slightly larger than that of the MDC and this is because the encoder did not take into account the second vector component of the second cluster in the MDC case. Multi-stage layered code outperforms the MDC when the number of received packets is between 22 and 38.

As a final remark, we can see from Fig. 4.12 that, for the bit-plane layered code, Mohr’s algorithm just gives equal loss protection (ELP) instead of unequal loss protection (ULP). Equal loss protection (ELP) manifests itself as a staircase plot with
only one step up while for ULP the SDR performance improves gradually through two steps as more packets are received. For the multi-stage layered code, Mohr’s algorithm gives two layers of protection as can be seen from the multi-stage layered curve. In this case 23 streams are given 21 bytes of data and 26 of them are given 22 bytes of data. Since for layers with different levels of FEC protection SDR gradually increases for ULP, the closer the Mohr’s allocation to ULP the better result would be achieved in comparison with MDC. So let us compare the profile of the two layered codes to see what makes one of them ULP and the other ELP. Fig. 4.13 shows the profile of these two algorithms in one graph as functions of number of received bytes. Fig. 4.14 depicts the profile of a 512 × 512 gray scale Lenna image encoded using SPHIT algorithm. We have chosen this profile because the output of Mohr’s algorithm with this input is composed of many layers or say it is highly ULP, and we want to see how ULP property manifests itself in the SDR plot. As can be seen from Fig. 4.13, the slope of the multi-stage profile at the beginning is larger than that of the bit-plane and from Fig. 4.14 the slope of the curve at the beginning is much larger than both the multi-stage and bit-plane layered codes. This is the cause for the ULP property of the Mohr’s output. So, we should look for layering approaches that can provide us with bigger slopes at the beginning. To this end, we propose two methods: (1) change the arrangement of the source output in such a way that the source vectors with more significant effect on the distortion first enter the encoder, (2) classify the points. Method (2) can be performed by computing wavelet coefficients, as in SPHIT, or using other transforms.

In Fig. 4.12, we compared three multiple description coding algorithms. Two of them namely multi-stage layered code, and bit-plane layered code are based on
CHAPTER 4. SIMULATION RESULTS

Figure 4.13: SDR in dB as functions of number of received bytes for multi-stage and multi-plane layered streams

Figure 4.14: PSNR in dB as function of number of received bytes for SPHIT

FEC and one is the MMMDC described in Chapter 2. As the figure shows, the MD algorithms that are based on FEC still suffer from the cliff effect. That is, unlike MMMDC, their performances in terms of SDR do not improve smoothly as more packets are received. The MMMDC curve would have had sharper initial shape if the bits were ordered differently. On the other hand, these FEC based MDC methods are more resilient against particular packet losses. That is, due to the MDS property of Reed Solomon codes, no matter which packets are lost, as long as enough number of
packets are received by the decoder, it is capable of decoding the received data. For MMMDC system this is not the case, and the plot is the mean SDR performance for 10 independent runs. In the following section, we are going to compare FEC-based MDC algorithm with MMMDC for speech data. For speech data, we will compare the two system performances to see the amount of resilience of FEC-based system against particular packet losses.

4.6 Multiple Description Coding on G.711.1 Speech Data

In this section we compare the two algorithms on G.711.1 speech data to enhance the error resilience of the coder using MDC.

As was discussed in Chapter 3, the encoded bit stream of G.711.1 can consist of different layers depending on the encoder mode of operation. There is always a base layer for the lower-band signal which can be followed by the lower-band enhancement layer, and another layer for the higher-band signal that we call second layer.

The higher-band layer of the encoder transforms higher-band signal into the MDCT domain, and encode MDCT coefficients by a CSVQ quantizer. Instead of using of CSVQ for the quantization of MDCT coefficients, we want to use multiple description codes. To this end, we replace G.711.1 higher-band encoder by an MD encoder. FEC-based MD encoder and mixture model-based MD encoder are to be compared for this purpose. So, MDCT coefficients extracted from the MDCT block are to be used as our data to compare FEC-based multiple description algorithm with MMMDC system. Because we want to work on the second layer, i.e., the layer that is
related to the higher-band signal, we have to work on the encoder modes that include layer 2 namely, $R_{2b}$ and $R_3$.

We reshape the MDCT coefficients of the speech signal into 2 dimensional vectors. Fig. 4.15 shows MDCT 2D points and contours of the Gaussian distribution that is found from the EM algorithm. The distribution is modeled with one Gaussian component in this figure. Since the change in log-likelihood in the EM algorithm from 1 component to 2 components is negligible for this data, it is reasonable to model the data with one Gaussian component only. Thus, we have 1 cluster with two vector components for each data vector and we need two MDSQs to encode the data.

Each coefficient in layer 2 is encoded with 2 bits in G.711.1, so, we have 4 bits for each 2D vector to allocate. Because, two eigenvalues of the estimated Gaussian distribution are almost the same for our data, and bit allocation among vector components depends on the eigenvalues, the allocation formula would give us the same
number of bits for two vector components. So, the side rate for each MDSQ would be 
\( R = 1 \) bit.

For comparison between FEC-based MDC and MDC based on index assignment, we have trained our MDSQs with the whole transformed data set. Before doing the training, KLT transform is implemented on data, and the transformed data is used for training MDSQs. To be more practical, MDSQs should be trained with unit-variance Gaussian distribution because KLT transform has normalized data variance; however, this would make both systems perform with less reconstructed quality, because after all Gaussian distribution is only the estimate of data distribution.

In order to packetize encoded data, let us assume that the packet size is 35 bytes, so each packet would contain 140 2D-vectors. The first received packet is side 1 of the first 140 vectors, the second received packet is side 2 of the first 140 vectors, and the third received packet is side 1 of the second 140 vectors, and so forth. Thus, after receiving four consecutive packets the decoder would be able to decode 280 vectors with the best quality, because it has received both sides of those 280 vectors without erasures. The problem with this kind of packetization is for channels with burst erasures like fading channels with deep fades for more than one packet length. To make the system resilient against this case, we can use pseudo random generators at the encoder to interleave packets and send them to the channel. At the decoder, the order should be recovered first by using a random generator that is synchronized with the encoder one.

Now, let us investigate the FEC-based MD algorithm on G.711.1 MDCT data. Input to the Mohr’s algorithm is a layered code. We can produce layers using embedded quantization before generating multiple descriptions with Mohr’s algorithm. To
have a fair comparison between the two MDC approaches, we should have the same channel uses for both of them. Side rate for each MDCT 2D vector was 2 bits for the MMMDC system. This leads us to the central rate of 4 bits for each MDCT vector in index assignment based MDC. To have the same number of packets in FEC-based approach as the index assignment, we have to encode each MDCT coefficient with less than 4 bits to let some space for channel code redundancy. Another issue that is worth investigating is how to allocate bits for each layer. For instance, if we choose 3 bits for each vector, we can allocate 1 bit to each layer and have three layers, or we can have two layers one with 2 bits and the other with 1 bit. For now, let us have 2 bits for each vector and two layers each with 1 bit. With the total of two bits for two layers, we can quantize data with a 4 level Lloyd-Max quantizer. The first layer can distinguish two levels with 1 bit, so, it can specify which side of the mean each MDCT coefficient belongs to, and the second layer would specify the exact interval between two intervals at that side.

The layered code is given to the Mohr’s algorithm as input, and the algorithm generates descriptions. Each output packet from the output of the algorithm can be regarded as one description of the data. Because our embedded input bit stream consists of two layers, Mohr’s algorithm would create two levels of protection with more protection for the first layer.

The advantage of FEC-based MDC over MDC based on index assignment is its resilience against specific packet erasures. The former performance only relies on the number of lost packets while for the latter the reconstruction quality changes considerably for the same number of lost packets depending on which packets are being lost.
Fig. 4.16 compares these two systems for a channel with binomial loss profile with mean loss probability of 0.2. The figure shows reconstructed quality in dB as functions of number of lost packets. For MDC based on index assignment, for each number of lost packets, we have chosen erased packets at random and compute the reconstructed quality. This procedure is implemented 200 times for each number of lost packets to investigate the sensitivity of the system over specific packet erasures. For every number of lost packets, for each run, the SDR is shown with a dot in the figure resulting in a vertical line for every number of lost packets. Star (“*”) points show the performance of FEC-based MDC. It can be seen that for small numbers of packet losses, index assignment quality can change up to 10 dB depending on specific packet erasures, and it still outperforms that of FEC-based. For packet losses from 140 to 300, FEC-based quality is better. For 140 packet erasures, it might be better to use FEC-based than index assignment, because at half times the quality of index assignment is worse. While on occasions index assignment approach can give better quality, FEC-based is better optimized for channel conditions, because the probability that we have erasures between 140 and 300 is 0.97 for this channel profile. This means that FEC-based MDC is a better choice 97% of times.
Figure 4.16: FEC-based MDC vs MMMDC for higher-band G.711.1
Chapter 5

Summary and Conclusions

5.1 Summary

We first reviewed basic ideas in source coding and channel coding. Basically, there are two methods to take advantage of coding in a communication system. The first method is to design source encoder without taking into account channel failures with a goal to compress the source as far as it is possible, and design channel encoder in a way to add controlled amount of redundancy to the encoded bit stream to combat channel failures this method is called tandem source-channel coding in the literature. The second method is joint source-channel coding in which source information is compressed in a way to minimize the conditional distortion with a condition that is specified by channel transition matrix. It was pointed out that, although theoretically tandem source-channel coding must lead to the same result as joint source-channel coding, due to communication constraints in practical applications like delay and computational complexity, joint source-channel coding outperforms tandem source-channel coding.
Next, we reviewed multiple description coding, and reviewed different methods in the literature to do multiple description coding as well as rate distortion bounds for multiple description coding. With some modifications to the multiple description coding problem formulation, we observed that it is possible to investigate multiple description coding as a joint source-channel coding method.

In Chapter 2, we reviewed mixture models to estimate unknown source pdf, and mixture model-based quantization. We reviewed mixture model-based multiple description coding (MM-MDC). In that chapter, we analyzed the performance of MM-MDC under the high rate assumption. Through our asymptotic analysis, we ended up with redundancy and rate allocation formulations for MM-MDC systems with general mixture models for both fixed and variable rate cases. We also analyzed the system performance over packet erasure channels, and investigated the effect of channel erasure probability on average central and side distortions. Redundancy allocation formulation, and the method that we proposed to find integer number of diagonals for index assignment tables of multiple description scalar quantizers, enabled us to design an optimal MM-MDC system with lower complexity than the previous design method which was based on finding index assignment tables through searching among various possibilities, and picking up the one that leads to the smallest total average distortion.

The goal of the thesis is to design an optimal MM-MDC system, and use it to enhance the error resilience of G.711.1 speech coder over packet erasure channels. To this end, in the third chapter, we reviewed the newly standardized G.711.1 speech coder both for the encoder and the decoder, and in Chapter 4, we first evaluated the
performance of the designed MMMDC for a data model with two Gaussian components, and compared performances of the two methods of multiple description coding namely FEC-based multiple description coding and MMMDC system for the data model. We then compared the two algorithms for higher-band MDCT coefficients.

5.2 Conclusions

Rate and redundancy allocation formulations are derived for variable and fixed length codes through asymptotic analysis of the MMMDC system. The rate allocation formulations are for general mixture models, and general orthogonal transform in the MMMDC system. Analysis results show that the same amount of redundancy should be assigned to all multiple description scalar quantizers for optimal system performance, and an algorithm is proposed to find the number of diagonals in index assignment tables of MDSQs.

The system performance through packet erasure channels is also analyzed by asymptotic analysis. The results show that erasure probability of the channel specifies the amount of redundancy for the MDSQs, and rate allocations can be performed independent of the channel erasure probability. It is shown that the total average distortion is an increasing function of erasure probability \( p \). For erasure probabilities of less than 0.25 the effect of central distortion is dominant which is an increasing function of erasure probability. On the other hand, for erasure probabilities of greater than 0.25 the effect of side distortion is dominant which is a decreasing function of erasure probability. As channel erasure probability increases, less redundancy is assigned to index assignment tables. Thus, central distortion increases, and side distortion decreases. The simulation results show that the average total distortion that
is obtained from asymptotic analysis can closely predict the simulated total average
distortion, and the prediction becomes more valid for greater rates for which high
rate assumption is more valid.

The performance of the designed MMMDC system is compared with that of the
FEC-based multiple description coding system for Gaussian model data, and G.711.1
higher-band signals. Two layering approaches are proposed for the FEC-based algo-
rithm to produce a layered bit-stream that can be given to the Mohr algorithm for
FEC assignment. The results show that the FEC-based MD system suffers from the
“cliff effect” while the MMMDC performance improves smoothly as more packets are
received. On the other hand, the advantage of FEC-based MD system is its resilience
against specific packet losses. For the data model the simulated curve of the total
average distortion can be predicted and approximated by the asymptotic curve that
we derived via asymptotic analysis.

5.3 Future Work

1. GMM-MDC can be applied to quantize the deinterleaved MDCT coefficients in
G.711.1 and then compared with the CSVQ scheme native to G.711.1. The com-
parison is germane as CSVQ is a form of MDVQ [31],[32]. The six-dimensional
deinterleaved coefficient vectors might be better matched to multimodal mixture
densities.

2. Frame-based MDSQ/MDVQ does not exploit the long delay available to the
Mohr-FEC scheme. With long delay, larger transforms in conjunction with, for
instance, correlation-transform based MDSQ can be explored.
Bibliography


