

Ramachandran, R.P. “Quantization of Discrete Time Signals”
Digital Signal Processing Handbook
Ed. Vijay K. Madisetti and Douglas B. Williams
Boca Raton: CRC Press LLC, 1999

6

Quantization of Discrete Time Signals

- 6.1 [Introduction](#)
- 6.2 [Basic Definitions and Concepts](#)
Quantizer and Encoder Definitions • Distortion Measure •
Optimality Criteria
- 6.3 [Design Algorithms](#)
Lloyd-Max Quantizers • Linde-Buzo-Gray Algorithm
- 6.4 [Practical Issues](#)
- 6.5 [Specific Manifestations](#)
Multistage VQ • Split VQ
- 6.6 [Applications](#)
Predictive Speech Coding • Speaker Identification
- 6.7 [Summary](#)
- [References](#)

Ravi P. Ramachandran
Rowan University

6.1 Introduction

Signals are usually classified into four categories. A continuous time signal $x(t)$ has the field of real numbers $\mathbf{R}$ as its domain in that t can assume any real value. If the range of $x(t)$ (values that $x(t)$ can assume) is also $\mathbf{R}$, then $x(t)$ is said to be a continuous time, continuous amplitude signal. If the range of $x(t)$ is the set of integers $\mathbf{Z}$, then $x(t)$ is said to be a continuous time, discrete amplitude signal. In contrast, a discrete time signal $x(n)$ has $\mathbf{Z}$ as its domain. A discrete time, continuous amplitude signal has $\mathbf{R}$ as its range. A discrete time, discrete amplitude signal has $\mathbf{Z}$ as its range. Here, the focus is on discrete time signals. Quantization is the process of approximating any discrete time, continuous amplitude signal into one of a finite set of discrete time, continuous amplitude signals based on a particular distortion or distance measure. This approximation is merely signal compression in that an infinite set of possible signals is converted into a finite set. The next step of encoding maps the finite set of discrete time, continuous amplitude signals into a finite set of discrete time, discrete amplitude signals.

A signal $x(n)$ is quantized one block at a time in that p (almost always consecutive) samples are taken as a vector $\mathbf{x}$ and approximated by a vector $\mathbf{y}$. The signal or data vectors $\mathbf{x}$ of dimension p (derived from $x(n)$) are in the vector space $\mathbf{R}^p$ over the field of real numbers $\mathbf{R}$. Vector quantization is achieved by mapping the infinite number of vectors in $\mathbf{R}^p$ to a finite set of vectors in $\mathbf{R}^p$. There is an inherent compression of the data vectors. This finite set of vectors in $\mathbf{R}^p$ is encoded into another finite set of vectors in a vector space of dimension q over a finite field (a field consisting of a finite set of numbers). For communication applications, the finite field is the binary field $(0, 1)$. Therefore, the

original vector $\mathbf{x}$ is converted or compressed into a bit stream either for transmission over a channel or for storage purposes. This compression is necessary due to channel bandwidth or storage capacity constraints in a system.

The purpose of this chapter is to describe the basic definition and properties of vector quantization, introduce the practical aspects of design and implementation, and relate important issues. Note that two excellent review articles [1, 2] give much insight into the subject. The outline of the article is as follows. The basic concepts are elaborated on in Section 6.2. Design algorithms for scalar and vector quantizers are described in Section 6.3. A design example is also provided. The practical issues are discussed in Section 6.4. The multistage and split manifestations of vector quantizers are described in Section 6.5. In Section 6.6, two applications of vector quantization in speech processing are discussed.

6.2 Basic Definitions and Concepts

In this section, we will elaborate on the definitions of a vector and scalar quantizer, discuss some commonly used distance measures, and examine the optimality criteria for quantizer design.

6.2.1 Quantizer and Encoder Definitions

A quantizer, Q , is mathematically defined as a mapping [3] $Q : \mathbf{R}^p \rightarrow C$. This means that the p -dimensional vectors in the vector space $\mathbf{R}^p$ are mapped into a finite collection C of vectors that are also in $\mathbf{R}^p$. This collection C is called the codebook and the number of vectors in the codebook, N , is known as the codebook size. The entries of the codebook are known as codewords or codevectors. If $p = 1$, we have a scalar quantizer (SQ). If $p > 1$, we have a vector quantizer (VQ).

A quantizer is completely specified by p , C and a set of disjoint regions in $\mathbf{R}^p$ which dictate the actual mapping. Suppose C has N entries $\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N$. For each codevector, $\mathbf{y}_i$, there exists a region, R_i , such that any input vector $\mathbf{x} \in R_i$ gets mapped or quantized to $\mathbf{y}_i$. The region R_i is called a Voronoi region [3, 4] and is defined to be the set of all $\mathbf{x} \in \mathbf{R}^p$ that are quantized to $\mathbf{y}_i$. The properties of Voronoi regions are as follows:

1. Voronoi regions are convex subsets of $\mathbf{R}^p$.
2. $\bigcup_{i=1}^N R_i = \mathbf{R}^p$.
3. $R_i \cap R_j$ is the null set for $i \neq j$.

It is seen that the quantizer mapping is nonlinear and many to one and hence noninvertible.

Encoding the codevectors $\mathbf{y}_i$ is important for communications. The encoder, E , is mathematically defined as a mapping $E : C \rightarrow C_B$. Every vector $\mathbf{y}_i \in C$ is mapped into a vector $\mathbf{t}_i \in C_B$ where $\mathbf{t}_i$ belongs to a vector space of dimension $q = \lceil \log_2 N \rceil$ over the binary field $(0, 1)$. The encoder mapping is one to one and invertible. The size of C_B is also N . As a simple example, suppose C contains four vectors of dimension p , namely, $(\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3, \mathbf{y}_4)$. The corresponding mapped vectors in C_B are $\mathbf{t}_1 = [0\ 0]$, $\mathbf{t}_2 = [0\ 1]$, $\mathbf{t}_3 = [1\ 0]$ and $\mathbf{t}_4 = [1\ 1]$. The decoder D described by $D : C_B \rightarrow C$ performs the inverse operation of the encoder.

A block diagram of quantization and encoding for communications applications is shown in Fig. 6.1. Given that the final aim is to transmit and reproduce $\mathbf{x}$, the two sources of error are due to quantization and channel. The quantization error is $\mathbf{x} - \mathbf{y}_i$ and is heavily dealt with in this article. The channel introduces errors that transform $\mathbf{t}_i$ into $\mathbf{t}_j$ thereby reproducing $\mathbf{y}_j$ instead of $\mathbf{y}_i$ after decoding. Channel errors are ignored for the purposes of this article.

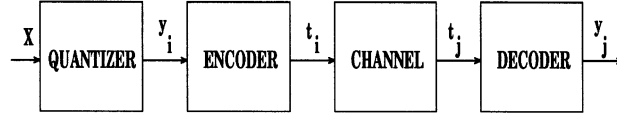


FIGURE 6.1: Block diagram of quantization and encoding for communication systems.

6.2.2 Distortion Measure

A distortion or distance measure between two vectors $\mathbf{x} = [x_1 \ x_2 \ x_3 \ \cdots \ x_p]^T \in \mathbf{R}^p$ and $\mathbf{y} = [y_1 \ y_2 \ y_3 \ \cdots \ y_p]^T \in \mathbf{R}^p$ where the superscript T denotes transposition is symbolically given by $d(\mathbf{x}, \mathbf{y})$. Most distortion measures satisfy three properties given by:

1. Positivity: $d(\mathbf{x}, \mathbf{y})$ is a real number greater than or equal to zero with equality if and only if $\mathbf{x} = \mathbf{y}$
2. Symmetry: $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$
3. Triangle inequality: $d(\mathbf{x}, \mathbf{z}) \leq d(\mathbf{x}, \mathbf{y}) + d(\mathbf{y}, \mathbf{z})$

To qualify as a valid measure for quantizer design, only the property of positivity needs to be satisfied. The choice of a distance measure is dictated by the specific application and computational considerations. We continue by giving some examples of distortion measures.

EXAMPLE 6.1: The L_r Distance

The L_r distance is given by

$$d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^p |x_i - y_i|^r \quad (6.1)$$

This is a computationally simple measure to evaluate. The three properties of positivity, symmetry, and the triangle inequality are satisfied. When $r = 2$, the squared Euclidean distance emerges and is very often used in quantizer design. When $r = 1$, we get the absolute distance. If $r = \infty$, it can be shown that [2]

$$\lim_{r \rightarrow \infty} d(\mathbf{x}, \mathbf{y})^{1/r} = \max_i |x_i - y_i| \quad (6.2)$$

This is the maximum absolute distance taken over all vector components.

EXAMPLE 6.2: The Weighted L_2 Distance

The weighted L_2 distance is given by:

$$d(\mathbf{x}, \mathbf{y}) = (\mathbf{x} - \mathbf{y})^T \mathbf{W} (\mathbf{x} - \mathbf{y}) \quad (6.3)$$

where $\mathbf{W}$ is the matrix of weights. For positivity, $\mathbf{W}$ must be positive-definite. If $\mathbf{W}$ is a constant matrix, the three properties of positivity, symmetry, and the triangle inequality are satisfied. In some applications, $\mathbf{W}$ is a function of $\mathbf{x}$. In such cases, only the positivity of $d(\mathbf{x}, \mathbf{y})$ is guaranteed to hold. As a particular case, if $\mathbf{W}$ is the inverse of the covariance matrix of $\mathbf{x}$, we get the Mahalanobis distance [2]. Other examples of weighting matrices will be given when we discuss the applications of quantization.

6.2.3 Optimality Criteria

There are two necessary conditions for a quantizer to be optimal [2, 3]. As before, the codebook C has N entries $\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N$ and each codevector $\mathbf{y}_i$ is associated with a Voronoi region R_i . The first condition known as the nearest neighbor rule states that a quantizer maps any input vector $\mathbf{x}$ to the codevector closest to it. Mathematically speaking, $\mathbf{x}$ is mapped to $\mathbf{y}_i$ if and only if $d(\mathbf{x}, \mathbf{y}_i) \leq d(\mathbf{x}, \mathbf{y}_j) \forall j \neq i$. This enables us to more precisely define a Voronoi region as:

$$R_i = \{\mathbf{x} \in \mathbf{R}^p : d(\mathbf{x}, \mathbf{y}_i) \leq d(\mathbf{x}, \mathbf{y}_j) \forall j \neq i\} \quad (6.4)$$

The second condition specifies the calculation of the codevector $\mathbf{y}_i$ given a Voronoi region R_i . The codevector $\mathbf{y}_i$ is computed to minimize the average distortion in R_i which is denoted by D_i where:

$$D_i = E[d(\mathbf{x}, \mathbf{y}_i) | \mathbf{x} \in R_i] \quad (6.5)$$

6.3 Design Algorithms

Quantizer design algorithms are formulated to find the codewords and the Voronoi regions so as to minimize the overall average distortion D given by:

$$D = E[d(\mathbf{x}, \mathbf{y})] \quad (6.6)$$

If the probability density $p(\mathbf{x})$ of the data $\mathbf{x}$ is known, the average distortion is [2, 3]

$$D = \int d(\mathbf{x}, \mathbf{y}) p(\mathbf{x}) d\mathbf{x} \quad (6.7)$$

$$= \sum_{i=1}^N \int_{R_i} d(\mathbf{x}, \mathbf{y}_i) p(\mathbf{x}) d\mathbf{x} \quad (6.8)$$

Note that the nearest neighbor rule has been used to get the final expression for D . If the probability density is not known, an empirical estimate is obtained by computing many sampled data vectors. This is called training data, or a training set, and is denoted by $T = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \dots, \mathbf{x}_M\}$ where M is the number of vectors in the training set. In this case, the average distortion is

$$D = \frac{1}{M} \sum_{k=1}^M d(\mathbf{x}_k, \mathbf{y}) \quad (6.9)$$

$$= \frac{1}{M} \sum_{i=1}^N \sum_{\mathbf{x}_k \in R_i} d(\mathbf{x}_k, \mathbf{y}_i) \quad (6.10)$$

Again, the nearest neighbor rule has been used to get the final expression for D .

6.3.1 Lloyd-Max Quantizers

The Lloyd-Max method is used to design scalar quantizers and assumes that the probability density of the scalar data $p(x)$ is known [5, 6]. Let the codewords be denoted by $y_1, y_2, \dots, y_N$. For each codeword y_i , the Voronoi region is a continuous interval $R_i = (v_i, v_{i+1}]$. Note that $v_1 = -\infty$ and $v_{N+1} = \infty$. The average distortion is

$$D = \sum_{i=1}^N \int_{v_i}^{v_{i+1}} d(x, y_i) p(x) dx \quad (6.11)$$

Setting the partial derivatives of D with respect to v_i and y_i to zero gives the optimal Voronoi regions and codewords.

In the particular case when $d(x, y_i) = (x - y_i)^2$, it can be shown that [5] the optimal solution is

$$v_i = \frac{y_i + y_{i+1}}{2} \quad (6.12)$$

for $2 \leq i \leq N$ and

$$y_i = \frac{\int_{v_i}^{v_{i+1}} x p(x) dx}{\int_{v_i}^{v_{i+1}} p(x) dx} \quad (6.13)$$

for $1 \leq i \leq N$. The overall iterative algorithm is

1. Start with an initial codebook and compute the resulting average distortion.
2. Solve for v_i .
3. Solve for y_i .
4. Compute the resulting average distortion.
5. If the average distortion decreases by a small amount that is less than a given threshold, the design terminates. Otherwise, go back to Step 2.

The extension of the Lloyd-Max algorithm for designing vector quantizers has been considered [7]. One practical difficulty is whether the multidimensional probability density function $p(\mathbf{x})$ is known or must be estimated. Even if this is circumvented, finding the multidimensional shape of the convex Voronoi regions is extremely difficult and practically impossible for dimensions greater than 5 [7]. Therefore, the Lloyd-Max approach cannot be extended to multidimensions and methods have been configured to design a VQ from training data. We will now elaborate on one such algorithm.

6.3.2 Linde-Buzo-Gray Algorithm

The input to the Linde-Buzo-Gray (LBG) algorithm [7] is a training set $T = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \dots, \mathbf{x}_M\} \in \mathbf{R}^p$ having M vectors, a distance measure $d(\mathbf{x}, \mathbf{y})$, and the desired size of the codebook N . From these inputs, the codewords $\mathbf{y}_i$ are iteratively calculated. The probability density $p(\mathbf{x})$ is not explicitly considered and the training set serves as an empirical estimate of $p(\mathbf{x})$. The Voronoi regions are now expressed as:

$$R_i = \{\mathbf{x}_k \in T : d(\mathbf{x}_k, \mathbf{y}_i) \leq d(\mathbf{x}_k, \mathbf{y}_j) \quad \forall j \neq i\} \quad (6.14)$$

Once the vectors in R_i are known, the corresponding codevector $\mathbf{y}_i$ is found to minimize the average distortion in R_i as given by

$$D_i = \frac{1}{M_i} \sum_{\mathbf{x}_k \in R_i} d(\mathbf{x}_k, \mathbf{y}_i) \quad (6.15)$$

where M_i is the number of vectors in R_i . In terms of D_i , the overall average distortion D is

$$D = \sum_{i=1}^N \frac{M_i}{M} D_i \quad (6.16)$$

Explicit expressions for $\mathbf{y}_i$ depend on $d(\mathbf{x}, \mathbf{y}_i)$ and two examples are given. For the L_1 distance,

$$\mathbf{y}_i = \text{median} [\mathbf{x}_k \in R_i] \quad (6.17)$$

For the weighted L_2 distance in which the matrix of weights $\mathbf{W}$ is constant,

$$\mathbf{y}_i = \frac{1}{M_i} \sum_{\mathbf{x}_k \in R_i} \mathbf{x}_k \quad (6.18)$$

which is merely the average of the training vectors in R_i . The overall methodology to get a codebook of size N is

1. Start with an initial codebook and compute the resulting average distortion.
2. Find R_i .
3. Solve for $\mathbf{y}_i$.
4. Compute the resulting average distortion.
5. If the average distortion decreases by a small amount that is less than a given threshold, the design terminates. Otherwise, go back to Step 2.

If N is a power of 2 (necessary for coding), a growing algorithm starting with a codebook of size 1 is formulated as follows:

1. Find codebook of size 1.
2. Find initial codebook of double the size by doing a binary split of each codevector. For a binary split, one codevector is split into two by small perturbations.
3. Invoke the methodology presented earlier of iteratively finding the Voronoi regions and codevectors to get the optimal codebook.
4. If the codebook of the desired size is obtained, the design stops. Otherwise, go back to Step 2 in which the codebook size is doubled.

Note that with the growing algorithm, a locally optimal codebook is obtained. Also, scalar quantizer design can also be performed.

Here, we present a numerical example in which $p = 2$, $M = 4$, $N = 2$, $T = \{\mathbf{x}_1 = [0 \ 0], \mathbf{x}_2 = [0 \ 1], \mathbf{x}_3 = [1 \ 0], \mathbf{x}_4 = [1 \ 1]\}$, and $d(\mathbf{x}, \mathbf{y}) = (\mathbf{x} - \mathbf{y})^T (\mathbf{x} - \mathbf{y})$. The codebook of size 1 is $\mathbf{y}_1 = [0.5 \ 0.5]$. We will invoke the LBG algorithm twice, each time using a different binary split. For the first run:

1. Binary split: $\mathbf{y}_1 = [0.51 \ 0.5]$ and $\mathbf{y}_2 = [0.49 \ 0.5]$.
2. Iteration 1
 - (a) $R_1 = \{\mathbf{x}_3, \mathbf{x}_4\}$ and $R_2 = \{\mathbf{x}_1, \mathbf{x}_2\}$.
 - (b) $\mathbf{y}_1 = [1 \ 0.5]$ and $\mathbf{y}_2 = [0 \ 0.5]$.
 - (c) Average distortion: $D = 0.25[(0.5)^2 + (0.5)^2 + (0.5)^2 + (0.5)^2] = 0.25$.
3. Iteration 2
 - (a) $R_1 = \{\mathbf{x}_3, \mathbf{x}_4\}$ and $R_2 = \{\mathbf{x}_1, \mathbf{x}_2\}$.
 - (b) $\mathbf{y}_1 = [1 \ 0.5]$ and $\mathbf{y}_2 = [0 \ 0.5]$.
 - (c) Average distortion: $D = 0.25[(0.5)^2 + (0.5)^2 + (0.5)^2 + (0.5)^2] = 0.25$.
4. No change in average distortion, the design terminates.

For the second run:

1. Binary split: $\mathbf{y}_1 = [0.5 \ 0.51]$ and $\mathbf{y}_2 = [0.5 \ 0.49]$.
2. Iteration 1
 - (a) $R_1 = \{\mathbf{x}_2, \mathbf{x}_4\}$ and $R_2 = \{\mathbf{x}_1, \mathbf{x}_3\}$.
 - (b) $\mathbf{y}_1 = [0.5 \ 1]$ and $\mathbf{y}_2 = [0.5 \ 0]$.

(c) Average distortion: $D = 0.25[(0.5)^2 + (0.5)^2 + (0.5)^2 + (0.5)^2] = 0.25$.

3. Iteration 2

(a) $R_1 = \{\mathbf{x}_2, \mathbf{x}_4\}$ and $R_2 = \{\mathbf{x}_1, \mathbf{x}_3\}$.

(b) $\mathbf{y}_1 = [0.5 \ 1]$ and $\mathbf{y}_2 = [0.5 \ 0]$.

(c) Average distortion: $D = 0.25[(0.5)^2 + (0.5)^2 + (0.5)^2 + (0.5)^2] = 0.25$.

4. No change in average distortion, the design terminates.

The two codebooks are equally good locally optimal solutions that yield the same average distortion. The initial condition as determined by the binary split influences the final solution.

6.4 Practical Issues

When using quantizers in a real environment, there are many practical issues that must be considered to make the operation feasible. First we enumerate the practical issues and then discuss them in more detail. Note that the issues listed below are interrelated.

1. Parameter set
2. Distortion measure
3. Dimension
4. Codebook storage
5. Search complexity
6. Quantizer type
7. Robustness to different inputs
8. Gathering of training data

A parameter set and distortion measure are jointly configured to represent and compress information in a meaningful manner that is highly relevant to the particular application. This concept is best illustrated with an example. Consider linear predictive (LP) analysis [8] of speech that is performed by the autocorrelation method. The resulting minimum phase nonrecursive filter

$$A(z) = 1 - \sum_{k=1}^p a_k z^{-k} \quad (6.19)$$

removes the near-sample redundancies in the speech. The filter $1/A(z)$ describes the spectral envelope of the speech. The information regarding the spectral envelope as contained in the LP filter coefficients a_k must be compressed (quantized) and coded for transmission. This is done in predictive speech coders [9]. There are other parameter sets that have a one-to-one correspondence to the set a_k . An equivalent parameter set that can be interpreted in terms of the spectral envelope is desired. The line spectral frequencies (LSFs) [10, 11] have been found to be the most useful.

The distortion measure is significant for meaningful quantization of the information and must be mathematically tractable. Continuing the above example, the LSFs must be quantized such that the spectral distortion between the spectral envelopes they represent is minimized. Mathematical tractability implies that the computation involved for (1) finding the codevectors given the Voronoi regions (as part of the design procedure) and (2) quantizing an input vector with the least distortion given a codebook is small. The L_1 , L_2 , and weighted L_2 distortions are mathematically feasible. For quantizing LSFs, the L_2 and weighted L_2 distortions are often used [12, 13, 14]. More details on LSF quantization will be provided in a forthcoming section on applications. At this point, a

general description is provided just to illustrate the issues of selecting a parameter set and a distortion measure.

The issues of dimension, codebook storage, and search complexity are all related to computational considerations. A higher dimension leads to an increase in the memory requirement for storing the codebook and in the number of arithmetic operations for quantizing a vector given a codebook (search complexity). The dimension is also very important in capturing the essence of the information to be quantized. For example, if speech is sampled at 8 kHz, the spectral envelope consists of 3 to 4 formants (vocal tract resonances) which must be adequately captured. By using LSFs, a dimension of 10 to 12 suffices for capturing the formant information. Although a higher dimension leads to a better description of the fine details of the spectral envelope, this detail is not crucial for speech coders. Moreover, this higher dimension imposes more of a computational burden. The codebook storage requirement depends on the codebook size N . Obviously, a smaller value of N imposes less of a memory requirement. Also for coding, the number of bits to be transmitted should be minimized, thereby diminishing the memory requirement. The search complexity is directly related to the codebook size and dimension. However, it is also influenced by the type of distortion measure.

The type of quantizer (scalar or vector) is dictated by computational considerations and the robustness issue (discussed later). Consider the case when a total of 12 bits are used for quantization, the dimension is 6, and the L_2 distance measure is utilized. For a VQ, there is one codebook consisting of $2^{12} = 4096$ codevectors each having 6 components. A total of $4096 \times 6 = 24576$ numbers need to be stored. Computing the L_2 distance between an input vector and one codevector requires 6 multiplications and 11 additions. Therefore, searching the entire codebook requires $6 \times 4096 = 24576$ multiplications and $11 \times 4096 = 45056$ additions. For an SQ, there are six codebooks, one for each dimension. Each codebook requires 2 bits or $2^2 = 4$ codewords. The overall codebook size is $4 \times 6 = 24$. Hence, a total of 24 numbers needs to be stored. Consider the first component of an input vector. Four multiplications and four additions are required to find the best codeword. Hence, for all 6 components, 24 multiplications and 24 additions are needed to complete the search. The storage and search complexity are always much less for an SQ.

The quantizer type is also closely related to the robustness issue. A quantizer is said to be robust to different test input vectors if it can maintain the same performance for a large variety of inputs. The performance of a quantizer is measured as the average distortion resulting from the quantization of a set of test inputs. A VQ takes advantage of the multidimensional probability density of the data as empirically estimated by the training set. An SQ does not consider the correlations among the vector components as a separate design is performed for each component based on the probability density of that component. For test data having a similar density to the training data, a VQ will outperform an SQ given the same overall codebook size. However, for test data having a density that is different from that of the training data, an SQ will outperform a VQ given the same overall codebook size. This is because an SQ can accomplish a better coverage of a multidimensional space. Consider the example in Fig. 6.2. The vector space is of two dimensions ($p = 2$). The component x_1 lies in the range 0 to $x_1(max)$ and x_2 lies between 0 and $x_2(max)$. The multidimensional probability density function (pdf) $p(x_1, x_2)$ is shown as the region ABCD in Fig. 6.2. The training data will represent this pdf and can be used to design a vector and scalar quantizer of the same overall codebook size. The VQ will perform better for test data vectors in the region ABCD. Due to the individual ranges of the values of x_1 and x_2 , the SQ will cover the larger space OKLM. Therefore, the SQ will perform better for test data vectors in OKLM but outside ABCD. An SQ is more robust in that it performs better for data with a density different from that of the training set. However, a VQ is preferable if the test data is known to have a density that resembles that of the training set.

In practice, the true multidimensional pdf of the data is not known as the data may emanate from many different conditions. For example, LSFs are obtained from speech material derived from many environmental conditions (like different telephones and noise backgrounds). Although getting a training set that is representative of all possible conditions gives the best estimate of the

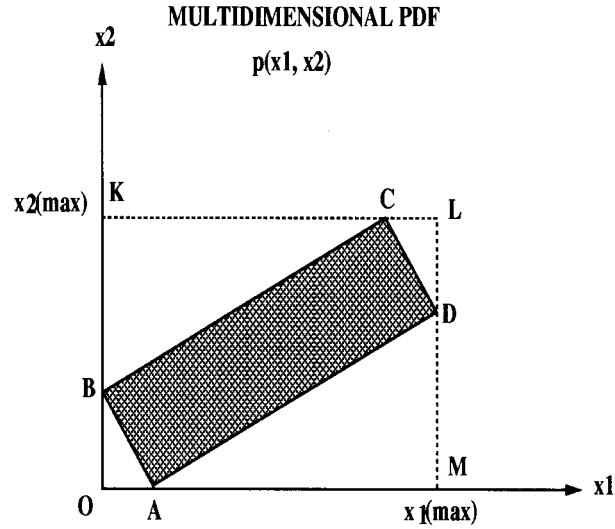


FIGURE 6.2: Example of a multidimensional probability density for explanation of the robustness issue.

multidimensional pdf, it is impossible to configure such a set in practice. A versatile training set contributes to the robustness of the VQ but increases the time needed to accomplish the design.

6.5 Specific Manifestations

Thus far, we have considered the implementation of a VQ as being a one-step quantization of $\mathbf{x}$. This is known as full VQ and is definitely the optimal way to do quantization. However, in applications such as LSF coding, quantizers between 25 and 30 bits are used. This leads to a prohibitive codebook size and search complexity. Two suboptimal approaches are now described that use multiple codebooks to alleviate the memory and search complexity requirements.

6.5.1 Multistage VQ

In multistage VQ consisting of R stages [3], there are R quantizers, $Q_1, Q_2, \dots, Q_R$. The corresponding codebooks are denoted as $C_1, C_2, \dots, C_R$. The sizes of these codebooks are $N_1, N_2, \dots, N_R$. The overall codebook size is $N = N_1 + N_2 + \dots + N_R$. The entries of the i th codebook C_i are $\mathbf{y}_1^{(i)}, \mathbf{y}_2^{(i)}, \dots, \mathbf{y}_{N_i}^{(i)}$. Figure 6.3 shows a block diagram of the entire system.

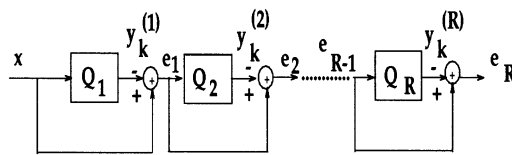


FIGURE 6.3: Multistage vector quantization.

The procedure for multistage VQ is as follows. The input $\mathbf{x}$ is first quantized by Q_1 to $\mathbf{y}_k^{(1)}$. The quantization error is $\mathbf{e}_1 = \mathbf{x} - \mathbf{y}_k^{(1)}$, which is in turn quantized by Q_2 to $\mathbf{y}_k^{(2)}$. The quantization error at the second stage is $\mathbf{e}_2 = \mathbf{e}_1 - \mathbf{y}_k^{(2)}$. This error is quantized at the third stage. The process repeats and at the R th stage, $\mathbf{e}_{R-1}$ is quantized by Q_R to $\mathbf{y}_k^{(R)}$ such that the quantization error is $\mathbf{e}_R$. The original vector $\mathbf{x}$ is quantized to $\mathbf{y} = \mathbf{y}_k^{(1)} + \mathbf{y}_k^{(2)} + \cdots + \mathbf{y}_k^{(R)}$. The overall quantization error is $\mathbf{x} - \mathbf{y} = \mathbf{e}_R$.

The reduction in the memory requirement and search complexity is best illustrated by a simple example. A full VQ of 30 bits will have one codebook of 2^{30} codevectors (cannot be used in practice). An equivalent multistage VQ of $R = 3$ stages will have three 10-bit codebooks C_1 , C_2 , and C_3 . The total number of codevectors to be stored is 3×2^{10} , which is practically feasible. It follows that the search complexity is also drastically reduced over that of a full VQ.

The simplest way to train a multistage VQ is to perform sequential training of the codebooks. We start with a training set $T = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \dots, \mathbf{x}_M\} \in \mathbf{R}^p$ to get C_1 . The entire set T is quantized by Q_1 to get a training set for the next stage. The codebook C_2 is designed from this new training set. This procedure is repeated so that all the R codebooks are designed. A joint design procedure for multistage VQ has been recently developed in [15] but is outside the scope of this article.

6.5.2 Split VQ

In split VQ [3], $\mathbf{x} = [x_1 \ x_2 \ x_3 \ \cdots \ x_p]^T \in \mathbf{R}^p$ is split or partitioned into R subvectors of smaller dimension as $\mathbf{x} = [\mathbf{x}^{(1)} \ \mathbf{x}^{(2)} \ \mathbf{x}^{(3)} \ \cdots \ \mathbf{x}^{(R)}]^T$. The i th subvector $\mathbf{x}^{(i)}$ has dimension d_i . Therefore, $p = d_1 + d_2 + \cdots + d_R$. Specifically,

$$\mathbf{x}^{(1)} = [x_1 \ x_2 \ \cdots \ x_{d_1}]^T \quad (6.20)$$

$$\mathbf{x}^{(2)} = [x_{d_1+1} \ x_{d_1+2} \ \cdots \ x_{d_1+d_2}]^T \quad (6.21)$$

$$\mathbf{x}^{(3)} = [x_{d_1+d_2+1} \ x_{d_1+d_2+2} \ \cdots \ x_{d_1+d_2+d_3}]^T \quad (6.22)$$

and so forth.

There are R quantizers, one for each subvector. The subvectors $\mathbf{x}^{(i)}$ are individually quantized to $\mathbf{y}_k^{(i)}$ so that the full vector $\mathbf{x}$ is quantized to $\mathbf{y} = [\mathbf{y}_k^{(1)} \ \mathbf{y}_k^{(2)} \ \mathbf{y}_k^{(3)} \ \cdots \ \mathbf{y}_k^{(R)}]^T \in \mathbf{R}^p$. The quantizers are designed using the appropriate subvectors in the training set T . The extreme case of a split VQ is when $R = p$. Then, $d_1 = d_2 = \cdots = d_p = 1$ and we get a scalar quantizer.

The reduction in the memory requirement and search complexity is again illustrated by a similar example as for multistage VQ. Suppose the dimension $p = 10$. A full VQ of 30 bits will have one codebook of 2^{30} codevectors. An equivalent split VQ of $R = 3$ splits uses subvectors of dimensions $d_1 = 3$, $d_2 = 3$, and $d_3 = 4$. For each subvector, there will be a 10-bit codebook having 2^{10} codevectors.

Finally, note that split VQ is feasible if the distortion measure is separable in that

$$d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^R d(\mathbf{x}^{(i)}, \mathbf{y}_k^{(i)}) \quad (6.23)$$

This property is true for the L_r distance and for the weighted L_2 distance if the matrix of weights $\mathbf{W}$ is diagonal.

6.6 Applications

In this article, two applications of quantization are discussed. One is in the area of speech coding and the other is in speaker identification. Both are based on LP analysis of speech [8] as performed by the autocorrelation method. As mentioned earlier, the predictor coefficients, a_k , describe a minimum phase nonrecursive LP filter $A(z)$ as given by Eq. (6.19). We recall that the filter $1/A(z)$ describes the spectral envelope of the speech, which in turn gives information about the formants.

6.6.1 Predictive Speech Coding

In predictive speech coders, the predictor coefficients (or a transformation thereof) must be quantized. The main aim is to preserve the spectral envelope as described by $1/A(z)$ and, in particular, preserve the formants. The coefficients a_k are transformed into an LSF vector $\mathbf{f}$. The LSFs are more clearly related to the spectral envelope in that (1) the spectral sensitivity is local to a change in a particular frequency and (2) the closeness of two adjacent LSFs indicates a formant.

Ideally, LSFs should be quantized to minimize the spectral distortion (SD) given by

$$\text{SD} = \sqrt{\frac{1}{B} \int_R \left[10 \log \left(|A_q(e^{j2\pi f})|^2 / |A(e^{j2\pi f})|^2 \right) \right]^2 df} \quad (6.24)$$

where $A(\cdot)$ refers to the original LP filter, $A_q(\cdot)$ refers to the quantized LP filter, B is the bandwidth of interest, and R is the frequency range of interest. The SD is not a mathematically tractable measure and is also not separable if split VQ is to be used. A weighted L_2 measure is used in which $\mathbf{W}$ is diagonal and the i th diagonal element is $w(i)$ is given by [14]:

$$w(i) = \frac{1}{f_i - f_{i-1}} + \frac{1}{f_{i+1} - f_i} \quad (6.25)$$

where $\mathbf{f} = [f_1 \ f_2 \ f_3 \ \cdots \ f_p]^T \in \mathbf{R}^p$, f_0 is taken to be zero, and f_{p+1} is taken to be the highest digital frequency (π or 0.5 if normalized). Regarding this distance measure, note the following:

1. The LSFs are ordered ($f_{i+1} > f_i$) if and only if the LP filter $A(z)$ is minimum phase. This guarantees that $w(i) > 0$.
2. The weight $w(i)$ is high if two adjacent LSFs are close to each other. Therefore, more weight is given to regions in the spectrum having formants.
3. The weights are dependent on the input vector $\mathbf{f}$. This makes the computation of the codevectors using the LBG algorithm different from the case when the weights are constant. However, for finding the codevector given a Voronoi region, the average of the training vectors in the region is taken so that the ordering property is preserved.
4. Mathematical tractability and separability of the distance measure are obvious.

A quantizer can be designed from a training set of LSFs using the weighted L_2 distance. Consider LSFs obtained from speech that is lowpass filtered to 3400 Hz and sampled at 8 kHz. If there are additional highpass or bandpass filtering effects, some of the LSFs tend to migrate [16]. Therefore, a VQ trained solely on one filtering condition will not be robust to test data derived from other filtering conditions [16]. The solution in [16] to robustize a VQ is to configure a training set consisting of two main components. First, LSFs from different filtering conditions are gathered to provide a reasonable empirical estimate of the multidimensional pdf. Second, a uniformly distributed set of vectors provides for coverage of the multidimensional space (similar to what is accomplished by an SQ). Finally, multistage or split LSF quantizers are used for practical feasibility [13, 15, 16].

6.6.2 Speaker Identification

Speaker recognition is the task of identifying a speaker by his or her voice. Systems performing speaker recognition operate in different modes. A closed set mode is the situation of identifying a particular speaker as one in a finite set of reference speakers [17]. In an open set system, a speaker is either identified as belonging to a finite set or is deemed not to be a member of the set [17]. For speaker verification, the claim of a speaker to be one in a finite set is either accepted or rejected [18]. Speaker recognition can either be done as a text-dependent or text-independent task. The difference is that in the former case, the speaker is constrained as to what must be said, while in the latter case no constraints are imposed. In this article, we focus on the closed set, text-independent mode. The overall system will have three components, namely, (1) LP analysis for parameterizing the spectral envelope, (2) feature extraction for ensuring speaker discrimination, and (3) classifier for making a decision. The input to the system will be a speech signal. The output will be a decision regarding the identity of the speaker.

After LP analysis of speech is carried out, the LP predictor coefficients, a_k , are converted into the LP cepstrum. The cepstrum is a popular feature as it provides for good speaker discrimination. Also, the cepstrum lends itself to the L_2 or weighted L_2 distance that is simple and yet reflective of the log spectral distortion between two LP filters [19]. To achieve good speaker discrimination, the formants must be captured. Hence, a dimension of 12 is usually used.

The cepstrum is used to develop a VQ classifier [20] as shown in Fig. 6.4. For each speaker enrolled in the system, a training set is established from utterances spoken by that speaker. From the training

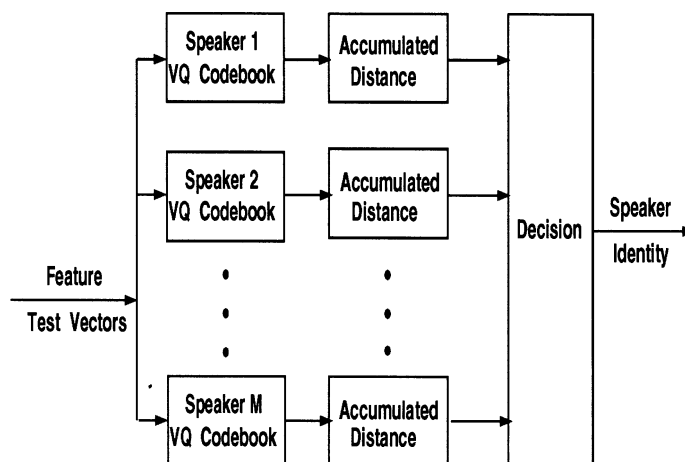


FIGURE 6.4: A VQ based classifier for speaker identification.

set, a VQ codebook is designed that serves as a speaker model. The VQ codebook represents a portion of the multidimensional space that is characteristic of the feature or cepstral vectors for a particular speaker. Good discrimination is achieved if the codebooks show little or no overlap as illustrated in Fig. 6.5 for the case of three speakers. Usually, a small codebook size of 64 or 128 codevectors is sufficient [21]. Even if there are 50 speakers enrolled, the memory requirement is feasible for real-time applications. An SQ is of no use because the correlations among the vector components are crucial for speaker discrimination. For the same reason, multistage or split VQ is also of no use. Moreover, full VQ can easily be used given the relatively smaller codebook size as compared to coding.

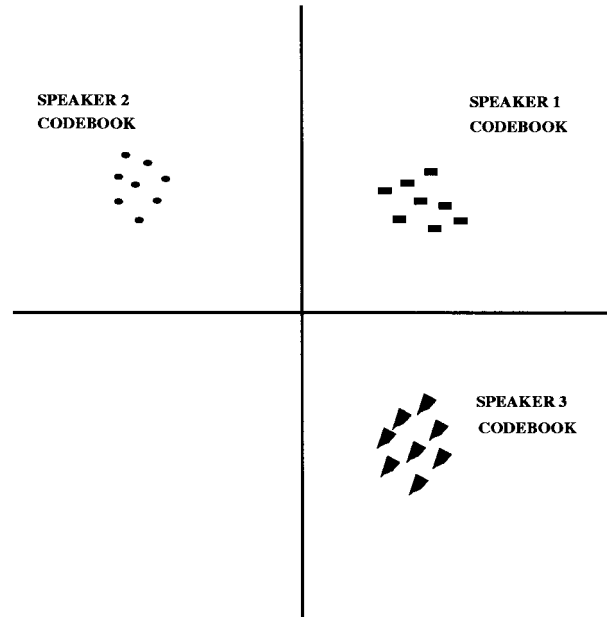


FIGURE 6.5: VQ codebooks for three speakers.

Given a random speech utterance, the testing procedure for identifying a speaker is as follows (see Fig. 6.4). First, the S test feature (cepstrum) vectors are computed. Consider the first vector. It is quantized by the codebook for speaker 1 and the resulting minimum L_2 or weighted L_2 distance is recorded. This quantization is done for all S vectors and the resulting minimum distances are accumulated (added up) to get an overall score for speaker 1. In this manner, an overall score is computed for all the speakers. The identified speaker is the one with the least overall score. Note that with the small codebook sizes, the search complexity is practically feasible. In fact, the overall score for the different speakers can be obtained in parallel. The performance measure for a speaker identification system is the identification success rate, which is the number of test utterances for which the speaker is identified correctly divided by the total number of test utterances.

The robustness issue is of great significance and emerges when the cepstral vectors derived from certain test speech material have not been considered in the training phase. This phenomenon of a full VQ not being robust to a variety of test inputs has been mentioned earlier and has been encountered in our discussion on LSF coding. The use of different training and testing conditions degrades performance since the components of the cepstrum vectors (such as LSFs) tend to migrate. Unlike LSF coding, appending the training set with a uniformly distributed set of vectors to accomplish coverage of a large space will not work as there will be much overlap among the codebooks of different speakers. The focus of the research is to develop more robust features that show little variation as the speech material changes [22, 23].

6.7 Summary

This article has presented a tutorial description of quantization. Starting from the basic definition and properties of vector and scalar quantization, design algorithms are described. Many practical aspects of design and implementation (such as distortion measure, memory, search complexity, and robustness) are discussed. These practical aspects are interrelated. Two important applications of vector quantization in speech processing are discussed in which these practical aspects play an important role.

References

- [1] Gray, R.M., Vector quantization, *IEEE Acoust. Speech Sig. Proc.*, 1, 4–29, Apr. 1984.
- [2] Makhoul, J., Roucos, S., and Gish, H., Vector quantization in speech coding, *Proc. IEEE*, 73, 1551–1588, Nov. 1985.
- [3] Gersho, A. and Gray, R.M., *Vector Quantization and Signal Compression*, Kluwer Academic Publishers, 1991.
- [4] Gersho, A., Asymptotically optimal block quantization, *IEEE Trans. Infor. Theory*, IT-25, 373–380, July 1979.
- [5] Jayant, N.S. and Noll, P., *Digital Coding of Waveforms, Principles and Applications to Speech and Video*, Prentice-Hall, Englewood Cliffs, NJ, 1984.
- [6] Max, J., Quantizing for minimum distortion, *IEEE Trans. Infor. Theory*, 7–12, Mar. 1960.
- [7] Linde, Y., Buzo, A., and Gray, R.M., An algorithm for vector quantizer design, *IEEE Trans. Comm.*, COM-28, 84–95, Jan. 1980.
- [8] Rabiner, L.R. and Schafer, R.W., *Digital Processing of Speech Signals*, Prentice-Hall, Englewood Cliffs, NJ, 1978.
- [9] Atal, B.S., Predictive coding of speech at low bit rates, *IEEE Trans. Comm.*, COM-30, 600–614, Apr. 1982.
- [10] Itakura, F., Line spectrum representation of linear predictor coefficients of speech signals, *J. Acoust. Soc. Amer.*, 57, S35(A), 1975.
- [11] Wakita, H., Linear prediction voice synthesizers: Line spectrum pairs (LSP) is the newest of several techniques, *Speech Technol.*, Fall 1981.
- [12] Soong, F.K. and Juang, B.-H., Line spectrum pair (LSP) and speech data compression, *IEEE Int. Conf. Acoust. Speech Signal Processing*, San Diego, CA, pp. 1.10.1–1.10.4, March 1984.
- [13] Paliwal, K.K. and Atal, B.S., Efficient vector quantization of LPC parameters at 24 bits/frame, *IEEE Trans. Speech Audio Processing*, 1, 3–14, Jan. 1993.
- [14] Laroia, R., Phamdo, N., and Farvardin, N., Robust and efficient quantization of speech LSP parameters using structured vector quantizers, *IEEE Intl. Conf. Acoust. Speech Signal Processing*, Toronto, Canada, 641–644, May 1991.
- [15] LeBlanc, W.P., Cuperman, V., Bhattacharya, B., and Mahmoud, S.A., Efficient search and design procedures for robust multi-stage VQ of LPC parameters for 4 kb/s speech coding, *IEEE Trans. Speech Audio Processing*, 1, 373–385, Oct. 1993.
- [16] Ramachandran, R.P., Sondhi, M.M., Seshadri, N., and Atal, B.S., A two codebook format for robust quantization of line spectral frequencies, *IEEE Trans. Speech Audio Processing*, 3, 157–168, May 1995.
- [17] Doddington, G.R., Speaker recognition—identifying people by their voices, *Proc. IEEE*, 73, 1651–1664, Nov. 1985.
- [18] Furui, S., Cepstral analysis technique for automatic speaker verification, *IEEE Trans. Acoust. Speech Sig. Proc.*, ASSP-29, 254–272, Apr. 1981.

- [19] Rabiner, L.R. and Juang, B.-H., *Fundamentals of Speech Recognition*, Prentice-Hall, Englewood Cliffs, NJ, 1993.
- [20] Rosenberg, A.E. and Soong, F.K., Evaluation of a vector quantization talker recognition system in text independent and text dependent modes, *Comp. Speech Lang.*, 22, 143–157, 1987.
- [21] Farrell, K.R., Mammone, R.J., and Assaleh, K.T., Speaker recognition using neural networks versus conventional classifiers, *IEEE Trans. Speech Audio Processing*, 2, 194–205, Jan. 1994.
- [22] Assaleh, K.T. and Mammone, R.J., New LP-derived features for speaker identification, *IEEE Trans. Speech Audio Processing*, 2, 630–638, Oct. 1994.
- [23] Zilovic, M.S., Ramachandran, R.P., and Mammone, R.J., Speaker identification based on the use of robust cepstral features derived from pole-zero transfer functions, accepted in *IEEE Trans. Speech Audio Processing*.