Lattice-Based Quantization Part II

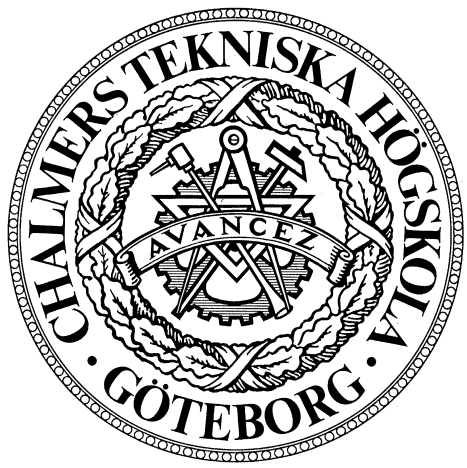
by

Thomas Eriksson and Erik Agrell

Lattice-Based Quantization, Part II

by

Thomas Eriksson and Erik Agrell
Department of Information Theory
Chalmers University of Technology
Göteborg, Sweden



Technical report no. 18
Department of Information Theory
Chalmers University of Technology
Göteborg, Sweden
Oct., 1996

ISSN 0283–1260

ABSTRACT

In this report we study vector quantization based on lattices. A lattice is an infinite set of points in a regular structure. The regularity can be exploited in vector quantization to make fast nearest-neighbor search possible, and to reduce the storage requirements. Aspects of lattice vector quantization, such as scaling and truncation of the infinite lattice, are treated. Theory for high rate lattice quantization is developed, and the performance of lattice quantization of Gaussian variables is investigated. We also propose a method to exploit the lattice regularity to design fast search algorithms for unconstrained vector quantization. Experiments on Gaussian input data illustrate that the method performs well in comparison to other fast search algorithms.

CONTENTS

1. INTRODUCTION

Vector quantization (VQ)[1] has since about 1980 become a popular technique for source coding of image and speech data. The popularity of VQ is motivated primarily by the theoretically optimal performance; no other source coding technique at equivalent delay can achieve better performance than optimal VQ. However, direct use of VQ suffers from a serious complexity barrier. Many authors have proposed constrained VQ structures to overcome the complexity, for example *multistage VQ* [1], *tree-structured VQ* [2-5], *vector-sum VQ* [6], *gain-shape VQ* [7], etc. Each of these solutions has disadvantages, in most cases a reduced performance. *Lattice VQ* [8, 9] is another constrained VQ technique, where the codevectors form a highly regular structure. The regular structure makes compact storage and fast *nearest-neighbor search* (finding the closest codevector to an input vector) possible, but also leads to performance loss.

Another line of research, also aimed to overcome the complexity barrier of VQ, is design of fast search methods for unconstrained quantizers. Due to the presumed lack of structure in such quantizers[2], nearest-neighbor search for unconstrained VQ is considerably more difficult than search of a constrained VQ. Algorithms for fast nearest-neighbor search of unconstrained VQ include for example *neighbor descent* methods [10, 11], where the complexity of a full search is avoided by precomputing an *adjacency table*, consisting of all neighbors to all VQ points. Other methods are the *anchor point* algorithm [12], where codevectors are excluded from the search by the triangle inequality, and the *K-d tree* technique [13], where a prestored tree structure helps in avoiding unnecessary operations.

In this report, we discuss *lattice-based quantization*[3] as a solution of the complexity problem. Lattice-based quantization is a generalization of conventional lattice quantization, by allowing modifications of the regular lattice structure while still maintaining a local lattice-similarity. In the first part of the report, conventional lattice quantization is treated. After the introduction and VQ preliminaries in chapter 1 and 2, we present high rate theory for lattice

[1]With VQ, we will sometimes mean *vector quantization*, and sometimes *vector quantizer*, with the distinction left to the context.

[2]A pdf-optimized unconstrained VQ is generally far from unstructured, but the structure may be difficult to find and exploit.

[3]Most of the conclusions in this report holds for *tessellation quantizers* as well. More about tessellations can be found in [14].

VQ for Gaussian variables in chapter 3. The high rate theory leads to design rules for lattice VQ, and formulas for asymptotic performance. Further, the performance of lattice VQ for a Gaussian input pdf is compared to the performance of pdf-optimized VQ. An important task in lattice VQ design is the *truncation* of an infinite-size lattice, to include the desired number of codevectors in the VQ. Other important aspects are for example the choice of lattice, and scaling of the source, to get a good performance. These aspects are treated from a practical perspective in chapter 3, and solutions are found, based on the lattice high rate theory. In many previous reports, the focus has been on high-dimensional lattice quantization, due to the *asymptotic equipartition property* (AEP); when the dimension grows to infinity, the *d*-dimensional probability density of a memoryless input source becomes more and more localized to a ”typical” region, inside which the density is approximately uniform [15]. Thus, a lattice quantizer, with an inherent uniform distribution of codevectors, can be expected to work well for high dimensions. We have instead focused on low-dimensional (2-5 dimensions) lattice VQ, since several interesting areas in speech and image coding employ low-dimensional parameter vectors.

The density of the codevectors in a lattice quantizer is uniform, which may inflict on the efficiency of lattice quantization for nonuniform sources. We propose a novel VQ design concept in chapter 4, with the goal to combine some of the desirable properties of a lattice VQ with the good performance of a pdf-optimized VQ. The VQ is initialized with a truncated lattice, and an adjacency table for the lattice is computed. Then, during the training, the quantizer is updated to keep the neighbors as given by the lattice adjacency table. By example, we show that this *lattice attraction* can be imposed with almost no performance loss at all for a Gaussian input pdf. A neighbor descent algorithm [11], modified to suit the special requirements of the lattice-attracted quantizers, is presented in chapter 5. The performance of the new neighbor descent method is reported in chapter 6, together with the performance of direct lattice quantization of Gaussian variables. Finally, a summary is given in chapter 7.

2. VECTOR QUANTIZATION

In this chapter, we present vector quantization theory. Necessary optimality conditions for a VQ is given, and theory for high rate quantization is discussed.

2.1 Definitions

A VQ Q of size N and dimension d is a mapping from a vector in the d-dimensional Euclidean space $\mathbb{R}^d$ into a finite reproduction set $\mathcal{C} = \{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_N\}$:

$$Q : \mathbb{R}^d \to \mathcal{C}. \tag{2.1}$$

The set $\mathcal{C}$, denoted the *codebook*, contains N *codevectors* $\mathbf{c}_k, k = 1, 2, ..., N$, each a vector in $\mathbb{R}^d$. The index k of the codevectors is denoted *codeword*. The rate R of the quantizer is defined as $\log_2(N)/d$ [bits per sample]. The definition of Q in (2.1) partitions $\mathbb{R}^d$ into N disjoint regions, each with a corresponding codevector $\mathbf{c}_k$.

The vector quantizer can be decomposed in two components, the encoder and the decoder. The encoder $\mathcal{E}$ maps from $\mathbb{R}^d$ to the index set $I = \{1, 2, ..., N\}$

$$\mathcal{E} : \mathbb{R}^d \to I, \tag{2.2}$$

and the decoder $\mathcal{D}$ maps the index set into the reproduction set $\mathcal{C}$, i.e.,

$$\mathcal{D} : I \to \mathbb{R}^d. \tag{2.3}$$

With this notation, the quantization operation can be written as a cascade of the encoder and decoder:

$$Q(\mathbf{x}) = \mathcal{D}(\mathcal{E}(\mathbf{x})). \tag{2.4}$$

In this report, we will measure the performance by the statistical mean of the squared Euclidean distance measure,

$$D = \mathrm{E}\left[\|\mathbf{x} - Q(\mathbf{x})\|^2\right]. \tag{2.5}$$

The mean squared error criterion is only one of many possible distortion measures, but it has the advantage of being widely used and is mathematically simple.

2.2 Optimality conditions

In VQ design, the aim is to find encoder and decoder rules to minimize the chosen distortion measure. For the squared Euclidean distance measure (2.5) (with a decoder $\mathcal{D}(i) = \mathbf{c}_i$), it can be readily shown [16] that for a fixed partition Ω_k of the input space, the codevectors $\{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_N\}$ should be chosen as the centroid of the vectors in the region,

$$\mathbf{c}_k = \mathrm{E}[\mathbf{x}|\mathbf{x} \in \Omega_k] \tag{2.6}$$

to minimize the expected distortion. (2.6) is often called *the centroid condition*. If instead the set of codevectors is fixed, the partition should be the *nearest neighbor partition*:

$$\Omega(\mathbf{c}_k) = \Omega_k = \left\{\mathbf{x} \in \mathbb{R}^d : \|\mathbf{x} - \mathbf{c}_k\|^2 \le \|\mathbf{x} - \mathbf{c}_i\|^2 \text{ for all } i \in I\right\} \tag{2.7}$$

with the corresponding encoder rule

$$\mathcal{E}(\mathbf{x}) = \underset{i \in I}{\operatorname{argmin}} \|\mathbf{x} - \mathbf{c}_i\|^2, \tag{2.8}$$

together with rules to solve ties. The regions Ω_k are often referred to as *Voronoi regions*, after the author of [17].

We see that both the encoder and the decoder are completely specified by the codebook $\mathcal{C}$, so finding optimal encoder and decoder rules is equivalent to finding the optimum set of codevectors $\{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_N\}$.

The centroid condition (2.6) and the nearest neighbor partition (2.7) are necessary but not sufficient for a VQ to be optimal in the mean square sense. Sufficient conditions for a globally optimal VQ have never been presented (except for some special cases), and a quantizer fulfilling the necessary conditions may be far from optimal. This makes VQ design a delicate problem.

Using the nearest neighbor condition, the *Voronoi neighbors* to a Voronoi region Ω_k in a VQ can be defined as

$$\mathcal{A}_k = \{i \in [1, N] : \Omega_i \cap \Omega_k \neq \varnothing\} \tag{2.9}$$

that is, the set of codevectors whose Voronoi regions share a face with Ω_k. With this definition, the nearest neighbor partition can be reformulated as

$$\Omega_k = \left\{\mathbf{x} \in \mathbb{R}^d : \|\mathbf{x} - \mathbf{c}_k\|^2 \le \|\mathbf{x} - \mathbf{c}_i\|^2 \text{ for all } i \in \mathcal{A}_k\right\}, \tag{2.10}$$

which illustrates that the Voronoi region is defined by a subset of the inequalities in (2.7). The new definition of the nearest neighbor partition shows that to find the optimum code-vector to a given input vector $\mathbf{x}$, it suffices to find a codevector whose Voronoi neighbors all have greater distance to the input vector. This can be exploited in fast search algorithms, as described in chapter 5.

2.3 High rate theory

In [18] and [16], it is shown that for high resolution VQs, the optimal reconstruction point density $\lambda(\mathbf{x})$ for quantization of a stochastic vector process $\mathbf{x}$ with pdf $f_{\mathbf{x}}(\mathbf{x})$ is given by

$$\lambda(\mathbf{x}) = a \cdot f_{\mathbf{x}}^{d/(d+2)}(\mathbf{x}) \tag{2.11}$$

where d is the dimension of the VQ, and a is a normalizing constant. For a quantizer with the above optimal point density, we have for high rates [16]

$$D \geq \frac{d \cdot \Gamma^{2/d}(d/2+1)}{(d+2)\cdot\pi}\left(\int f_{\mathbf{x}}(\mathbf{x})^{d/(d+2)}\right)^{(d+2)/d} \cdot 2^{-2R}, \tag{2.12}$$

where R is the rate of the quantizer, in bits per dimension.

For an uncorrelated Gaussian pdf, the above expression can be simplified to the Gaussian lower bound (GLB)

$$D_{\mathrm{GLB}} \geq 2^{-2R} \cdot f(d) \cdot \sigma_{\mathbf{x}}^2, \tag{2.13}$$

where

$$f(d) = \frac{2}{d}\left(\frac{d+2}{d}\right)^{d/2} \Gamma^{2/d}(d/2+1), \tag{2.14}$$

and

$$\sigma_{\mathbf{x}}^2 = \mathrm{E}\left[\|\mathbf{x}-\mathbf{m}_{\mathbf{x}}\|^2\right] = \int \|\mathbf{x}-\mathbf{m}_{\mathbf{x}}\|^2 f_{\mathbf{x}}(\mathbf{x})d\mathbf{x} \tag{2.15}$$

$$\mathbf{m}_{\mathbf{x}} = \mathrm{E}[\mathbf{x}] = \int \mathbf{x} \cdot f_{\mathbf{x}}(\mathbf{x})d\mathbf{x}. \tag{2.16}$$

Knagenhjelm [19] shows experimentally that the Gaussian lower bound is not only a lower bound, but also a good approximation to the actual performance of a well-trained vector quantizer, if the rate is high.

3. LATTICE QUANTIZATION

In this chapter, we will treat lattice quantization, both from a theoretical and a practical perspective. High rate theory for lattice quantization of iid Gaussian variables is derived, leading to formulas for lattice VQ design and performance. Practical issues in lattice VQ design, such as truncation and scaling of the lattice, are also treated.

3.1 Definitions

A *lattice* is an infinite set of points, defined as

$$\Lambda = \left\{ \mathbf{B}^T \cdot \mathbf{u} : \quad \mathbf{u} \in \mathbb{Z}^d \right\} \tag{3.1}$$

where $\mathbf{B}$ is the *generator matrix* of the lattice. The rows of $\mathbf{B}$ constitute a set of d linearly independent *basis vectors* for the lattice,

$$\mathbf{B} = \left[\mathbf{b}_1, \mathbf{b}_2, \cdots, \mathbf{b}_d \right]^T \tag{3.2}$$

Thus, the lattice Λ consists of all linear combinations of the basis vectors, with integer coefficients.

The *theta function* of the lattice gives the number of lattice points $\mathbf{c}_i$ at a specific distance from the origin, i.e. points within a *shell*. The theta function for many standard lattices can be found in [9].

The *fundamental parallelotope* of the lattice is defined as the parallelotope

$$z_1 \mathbf{b}_1 + ... + z_d \mathbf{b}_d \qquad (0 \le z_i < 1). \tag{3.3}$$

Associated with each lattice point is a Voronoi region. Due to the regular structure of lattices, all Voronoi regions in a lattice are simply translations of the Voronoi region $\Omega(\mathbf{0})$ around the zero lattice point. $\Omega(\mathbf{0})$ is referred to as the *lattice Voronoi region* Ω, with the definition

$$\Omega = \left\{ \mathbf{x} \in \mathbb{R}^d : \|\mathbf{x}\|^2 \le \|\mathbf{x} - \mathbf{c}\|^2 \text{ for all } \mathbf{c} \in \Lambda \right\} \tag{3.4}$$

The *normalized second moment* of a Voronoi region $\Omega(\mathbf{c}_i)$ is defined to be

$$G = \frac{1}{d}\left[\mathrm{vol}\left(\Omega(\mathbf{c}_i)\right)\right]^{-1-2/d} \int_{\Omega(\mathbf{c}_i)} \|\mathbf{x} - \mathbf{c}_i\|^2 d\mathbf{x}, \tag{3.5}$$

where $\mathrm{vol}\left(\Omega(\mathbf{c}_i)\right)$ is the volume of the Voronoi region around $\mathbf{c}_i$. Since $\Omega(\mathbf{c}_i)$ is a translation of Ω, $\Omega(\mathbf{c}_i) = \Omega + \mathbf{c}_i$, we can write

$$G = \frac{1}{d}\left[\mathrm{vol}(\Omega)\right]^{-1-2/d} \int_{\Omega} \|\mathbf{x}\|^2 d\mathbf{x}, \tag{3.6}$$

which illustrates that G is independent of i. The constant G is from now on be referred to as the *quantization constant* of the lattice, since it describes the mean squared error per dimension for quantization of an infinite uniform distribution, if the volume of the Voronoi region is normalized to one.

Lattice quantization is a special class of vector quantization, with the codebook having a highly regular structure. Any codevector $\mathbf{c}_k \in \mathcal{C}$ in a lattice quantizer can be written on the form

$$\mathbf{c}_k = \mathbf{B}^T \cdot \mathbf{u}_k \tag{3.7}$$

where $\mathbf{u}_k$ is one of N given integer vectors, and $\mathbf{B}$ is the generator matrix of the lattice. Alternatively, a lattice VQ can be described as the intersection between a *lattice* Λ and a *shape* $\mathcal{S}$,

$$\mathcal{C} = \Lambda \cap \mathcal{S} \tag{3.8}$$

where $\mathcal{S}$ is a d-dimensional bounded region in $\mathbb{R}^d$. An example is shown in figure 3.1.

Figure 3.1. Illustration of lattice truncation. Left: a lattice Λ, Center: a shape $\mathcal{S}$, Right: the resulting lattice quantizer $\mathcal{C}$.

The design of a lattice VQ can now be separated into finding a good lattice, specified through its generator matrix $\mathbf{B}$, and a good shape $\mathcal{S}$. In addition, a scale factor for the lattice must be found, and an assignment of indices to the codevectors. These problems will be treated in the following sections.

Applications of lattice vector quantization include, e.g., image coding [20, 21] and speech coding [22, 23]. Moayeri et al. superimposed a fine lattice upon a source-optimized

unstructured VQ to achieve a fast two-step search method [24, 25]. Kuhlmann and Bucklew [26], Swaszek [27] and Eriksson [28] connects lattices with different scaling into one "piecewise uniform" codebook, to approximate nonuniform source pdfs. In [14], an overview of applications including lattice VQ is presented.

3.2 Theory for high rate lattice quantization

In this section, we derive expressions for the distortion of lattice quantization of iid Gaussian vectors, when the rate *R* of the quantizer tends to infinity. Eyuboğlu and Forney [29], and Jeong and Gibson [30], have previously worked with high rate theory for lattice quantization, but to the authors' knowledge, simple analytical expressions for the optimal truncation and performance of *d*-dimensional lattice quantizers has not been presented before. A major difference between the high rate lattice theory presented here and the usual high rate theory for optimal quantization (section 2.3), is that for lattice quantization, it is necessary to explicitly consider overload distortion, while the usual high rate theory only permits granular distortion.

We assume an iid Gaussian input pdf, with zero mean, unit variance samples. However, in the end of this section we discuss a generalization of the results.

After some definitions, two theorems concerning the distortion of a lattice VQ as a function of the rate and truncation are given. The optimal truncation radius, and the corresponding distortion, are found by setting the derivative of the distortion to zero.

A *d-sphere* is a *d*-dimensional sphere, defined as

$$S_d(a) = \left\{ \mathbf{x} \in \mathbb{R}^d : \|\mathbf{x}\| \le a \right\}. \tag{3.9}$$

We assume a truncation shape in the form of a *d*-sphere with radius a_T (figure 3.2), so that

$$\mathcal{C} = (\Lambda - \mathbf{v}) \cap S_d(a_T), \tag{3.10}$$

where **v** is an arbitrary vector (see the discussion in section 3.4, and (3.33)).

We subdivide the *d*-dimensional space into two (nonspherical) subregions: a *granular* region $\mathcal{G}$, which we define as the union of lattice Voronoi regions around all codevectors,

$$\mathcal{G} = \bigcup_{\mathbf{c}_i \in \mathcal{C}} (\Omega + \mathbf{c}_i), \tag{3.11}$$

and an *overload* region $\overline{\mathcal{G}}$, which is the rest of the space, so that $\mathcal{G} \cup \overline{\mathcal{G}} = \mathbb{R}^d$ and $\mathcal{G} \cap \overline{\mathcal{G}} = \varnothing$. Figure 3.2 illustrates the granular and overload regions for a two-dimensional lattice VQ, based on the well-known hexagonal lattice A_2.

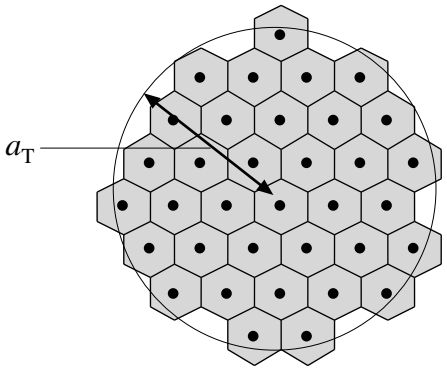


Figure 3.2. Illustration of the granular region (the gray area) and the overload region (everything but the gray area) of a 2-dimensional lattice quantizer.

The total distortion D of the lattice quantizer can be separated into a granular component, $D_{\mathcal{G}}$, and an overload component, $D_{\overline{\mathcal{G}}}$,

$$D = \int_{\mathbb{R}^d} \left\| \mathbf{x} - \mathbf{c}^* \right\|^2 f_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} = \int_{\mathcal{G}} \left\| \mathbf{x} - \mathbf{c}^* \right\|^2 f_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} + \int_{\overline{\mathcal{G}}} \left\| \mathbf{x} - \mathbf{c}^* \right\|^2 f_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} = D_{\mathcal{G}} + D_{\overline{\mathcal{G}}}, \quad (3.12)$$

where $\mathbf{c}^*$ denotes the codevector in the codebook $\mathcal{C}$ that is closest to the input vector $\mathbf{x}$. We now give two theorems, leading to simple approximations of the granular and the overload distortion of lattice quantization. In the first theorem, we write the overload distortion as the distortion given a high codevector density close to the surface of the truncation sphere, plus an error term. The second theorem is mainly based on the smoothness of the Gaussian pdf, so that the pdf within the granular Voronoi regions is nearly uniform, if the Voronoi regions are small. Both theorems are proved in appendix A.

Theorem I: The overload distortion is given by

$$D_{\overline{\mathcal{G}}} = f_{\overline{\mathcal{G}}}(d) \cdot a_T^{d-4} \cdot e^{-a_T^2/2} \cdot \left(1 + \varepsilon_{\overline{\mathcal{G}}}\right) \quad (3.13)$$

where $f_{\overline{\mathcal{G}}}(d) = \left(2^{d/2-2} \cdot \Gamma(d/2)\right)^{-1}$. For asymptotically high rates *R*, and the truncation radius a_T suitably chosen, $\varepsilon_{\overline{\mathcal{G}}}$ tends to zero.

Theorem II: The granular distortion is given by

$$D_{\mathcal{G}} = f_{\mathcal{G}}(d) \cdot a_T^2 \cdot 2^{-2R} \cdot \left(1 + \varepsilon_{\mathcal{G}}\right) \quad (3.14)$$

where $f_{\mathcal{G}}(d) = G \cdot d \cdot \pi \cdot \Gamma(d/2+1)^{-2/d}$. For asymptotically high rates *R*, and the truncation radius a_T suitably chosen, $\varepsilon_{\mathcal{G}}$ tends to zero.

The total distortion, *D*, can be written

$$D = D_{\mathcal{G}} + D_{\overline{\mathcal{G}}} = \left(f_{\mathcal{G}}(d) \cdot a_T^2 \cdot 2^{-2R} + f_{\overline{\mathcal{G}}}(d) \cdot a_T^{d-4} \cdot e^{-a_T^2/2} \right) \cdot (1 + \varepsilon), \quad (3.15)$$

where the error term ε tends to zero when *R* grows towards infinity. For the moment, we exclude the error term, and seek the minimum of

$$\hat{D} = \hat{D}_{\mathcal{G}} + \hat{D}_{\overline{\mathcal{G}}} = f_{\mathcal{G}}(d) \cdot a_{\mathrm{T}}^2 \cdot 2^{-2R} + f_{\overline{\mathcal{G}}}(d) \cdot a_{\mathrm{T}}^{d-4} \cdot e^{-a_{\mathrm{T}}^2/2}. \tag{3.16}$$

In appendix A.4, it is shown that the minimum value of $\hat{D}$ is also the minimum value of D. To find the value of the truncation radius a_{T} that minimizes the distortion, we differentiate $\hat{D}$ with respect to a_{T}:

$$\frac{\partial \hat{D}}{\partial a_{\mathrm{T}}} = 2 \cdot f_{\mathcal{G}}(d) \cdot a_{\mathrm{T}} \cdot 2^{-2R} + f_{\overline{\mathcal{G}}}(d) \cdot (d-4) \cdot a_{\mathrm{T}}^{d-5} \cdot e^{-a_{\mathrm{T}}^2/2} - f_{\overline{\mathcal{G}}}(d) \cdot a_{\mathrm{T}}^{d-3} \cdot e^{-a_{\mathrm{T}}^2/2}. \tag{3.17}$$

Since $\hat{D}$ is a convex and continuous function in the interesting region (see section A.4), we get the condition for minimal distortion by setting the derivative to zero,

$$\frac{\partial \hat{D}}{\partial a_{\mathrm{T}}} = 0 \quad \Leftrightarrow \quad f_{\overline{\mathcal{G}}}(d) \cdot a_{\mathrm{T,opt}}^{d-6} \cdot e^{-a_{\mathrm{T,opt}}^2/2} \cdot \left(a_{\mathrm{T,opt}}^2 + 4 - d \right) = 2 \cdot f_{\mathcal{G}}(d) \cdot 2^{-2R}. \tag{3.18}$$

where $a_{\mathrm{T,opt}}$ is the value of a_{T} that minimizes the distortion. We observe that by multiplying both sides of (3.18) with a_{T}^2, we get

$$\hat{D}_{\overline{\mathcal{G}}} \cdot \left(a_{\mathrm{T,opt}}^2 + 4 - d \right) = 2 \hat{D}_{\mathcal{G}} \tag{3.19}$$

where $\hat{D}_{\overline{\mathcal{G}}}$ and $\hat{D}_{\mathcal{G}}$ are given by (3.16). We get

$$\frac{\hat{D}_{\overline{\mathcal{G}}}}{\hat{D}_{\mathcal{G}}} = \frac{2}{a_{\mathrm{T,opt}}^2 + 4 - d}. \tag{3.20}$$

In appendix A it is shown that $a_{\mathrm{T,opt}}$ tends to infinity when R approaches infinity. We conclude that the total distortion is dominated by the granular distortion, when the rate tends to infinity,

$$\frac{D_{\overline{\mathcal{G}}}}{D_{\mathcal{G}}} \to 0 \text{ when } R \to \infty. \tag{3.21}$$

Returning to (3.18), and taking the logarithm of both sides, we have

$$-\frac{a_{\mathrm{T,opt}}^2}{2} + (d-6) \cdot \ln\left(a_{\mathrm{T,opt}} \right) + \ln\left(a_{\mathrm{T,opt}}^2 + 4 - d \right) = -R \cdot 2 \ln 2 + \ln\left(\frac{2 \cdot f_{\mathcal{G}}(d)}{f_{\overline{\mathcal{G}}}(d)} \right), \tag{3.22}$$

or, equivalently,

$$a_{\mathrm{T,opt}}^2 - (d-4) \cdot \ln\left(a_{\mathrm{T,opt}}^2 \right) - 2 \cdot \ln\left(1 + \frac{4-d}{a_{\mathrm{T,opt}}^2} \right) = 4 \ln 2 \cdot R - 2 \cdot \ln\left(\frac{2 \cdot f_{\mathcal{G}}(d)}{f_{\overline{\mathcal{G}}}(d)} \right). \tag{3.23}$$

Since $a_{\mathrm{T,opt}}$ tends to infinity for rates approaching infinity, both sides are dominated by their first terms, resulting in

$$a_{\mathrm{T,opt}}^2 \approx R \cdot 4 \ln 2 \text{ when } R \to \infty. \tag{3.24}$$

that is, the optimal truncation radius $a_{\mathrm{T,opt}}$ is proportional to the square root of R for asymptotically high rates.

The total distortion (3.15) can now be written

$$D = g(R,d)\cdot 2^{-2R}, \tag{3.25}$$

where $g(R,d)$ is approximated using (3.21) and (3.24),

$$g(R,d) \approx 4\cdot \ln 2 \cdot f_{\mathcal{G}}(d)\cdot R \quad \text{when } R\to\infty. \tag{3.26}$$

It is easy to generalize the formulas to arbitrary variance, by making the substitution $\mathbf{y} = \mathbf{x}\cdot\sqrt{\sigma_{\mathbf{y}}^2/d}$ (see (3.27)-(3.29)). If we compare the lattice VQ distortion with the distortion of a pdf-optimized quantizer (2.13), we see that the discrepancy increases with the rate. This can be observed in figure 3.7, section 3.6, where optimal VQ and lattice VQ are compared.

(3.25) is only proven for rates approaching infinity, but we have experimentally verified that the formulas also hold for realistic rates. In figure 3.3, the experimental performance of lattice quantization (see table 6.1) is compared to the high rate theory results, for quantization of 2- and 5-dimensional Gaussian variables.

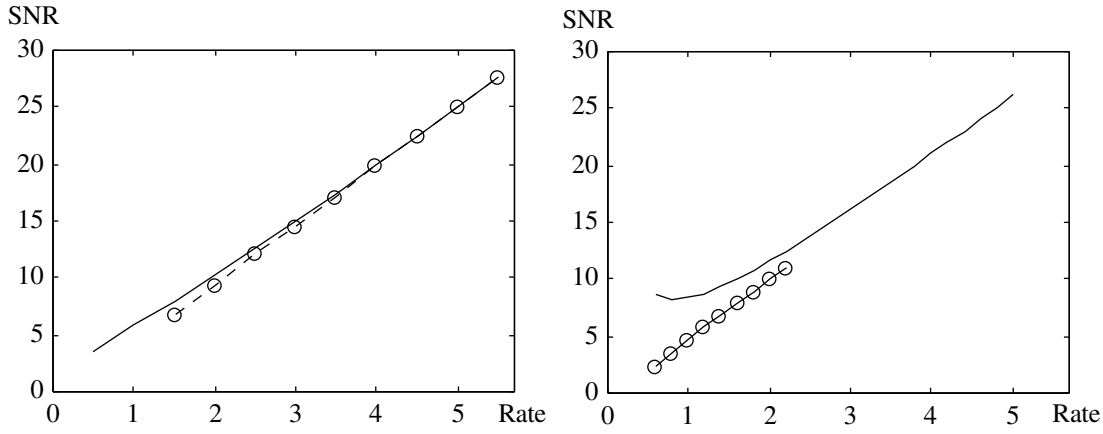


Figure 3.3. Experimental performance for lattice quantization of an iid Gaussian pdf (circles), and performance predicted by lattice VQ high rate theory (line). Left: 2 dimensions. Right: 5 dimensions.

With this theoretical derivation of lattice VQ performance, we have two asymptotical lattice VQ results: the asymptotic equipartition property predicts that a lattice VQ performs better for high dimensions, while the high rate theory predicts that a lattice VQ performs worse for high rates. These results are illustrated in figure 3.4, where each curve indicates a specific performance loss compared to a pdf-optimized VQ. The curves in figure 3.4 were computed by use of the high rate lattice theory (3.25) and the Gaussian high rate lower bound in (2.13).

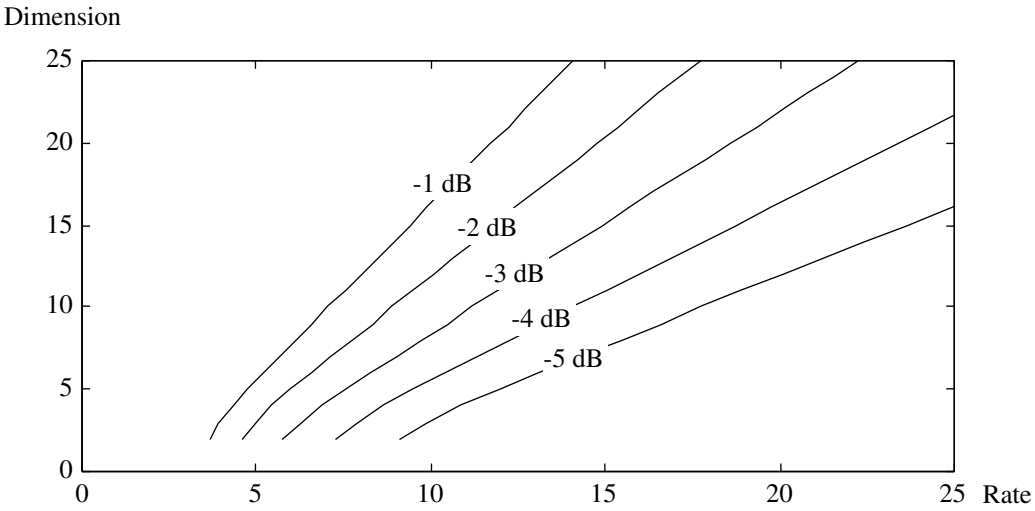


Figure 3.4. Estimated performance loss for a lattice VQ compared with a pdf-optimized VQ. The curves indicate rate and dimension for lattice quantizers with performance loss from 1 to 5 dB.

The formulas above were derived for iid Gaussian densities, with zero mean, unit variance samples, but it is straightforward to generalize the theory to arbitrary variance and mean. The conclusions should be similar also for correlated Gaussian data, but the theory is more complicated for correlated variables. By simple modifications, the formulas can be used for a generalized Gaussian pdf. Some of the results may also be possible to generalize to other pdfs. For all unbounded pdfs, such as Gaussian, Laplace, Gamma, etc., the size of the granular region must increase when the rate increases, for the overload distortion to be zero for an infinite rate. Thus, the granular region includes parts of the space with lower and lower pdf. Therefore, the larger the rate, the more the point density of an optimal quantizer, given by (2.11), differ from the uniform point density of a lattice quantizer. Based on the above reasoning, and on our experience of high rate theory for Gaussian pdfs, we believe that the suboptimality of lattice quantizers for high rates holds under far more general conditions than for iid Gaussian distributions.

Substituting as discussed above, to get formulas that are valid for arbitrary input signal variance, we conclude the high rate lattice theory in the following three points:

- The optimal squared truncation radius is proportional to the rate for high rates,

$$a_{\mathrm{T,opt}}^2 \approx R \cdot \frac{4 \ln 2}{d} \cdot \sigma_{\mathbf{y}}^2 \quad \text{when} \quad R \to \infty. \tag{3.27}$$

- For high rates, the granular distortion dominates over the overload distortion,

$$\frac{D_{\bar{G}}}{D_G} \to 0 \text{ when } R \to \infty. \tag{3.28}$$

- For high rates, the performance of lattice quantizers, as given by the high rate formula

$$D \approx R \cdot 2^{-2R} \cdot G \cdot 4 \cdot \ln 2 \cdot \pi \cdot \Gamma(d/2+1)^{-2/d} \cdot \sigma_{\mathbf{y}}^2 \text{ when } R \to \infty, \tag{3.29}$$

is inferior to the performance of optimal vector quantizers, given by the Gaussian lower bound (2.13).

3.3 Selection of lattice

The choice of lattice is of course of major importance for the performance of a lattice VQ. Ideally, the lattice should be selected to suit both the actual pdf and the truncation. However, for high rate quantization of smooth pdfs, the choice of lattice is fairly independent of input pdf and truncation [16]. For these cases, the lattice can be chosen based on its quantization performance for an infinite uniform pdf. This choice is motivated by high rate theory; for high rates, the pdf in each Voronoi region can be expected to be approximately uniform, at least for reasonably smooth pdfs (such as the Gaussian pdf). Further, the performance of infinite uniform lattice quantization, given by the quantization constant G, is easily found in the literature for many lattices.

Conway and Sloane [9] give values of the quantization constant G and lattice basis $\mathbf{B}$ for several lattices. For example, the best known lattices for quantization of infinite uniform pdfs in 2 and 5 dimensions are generated by, respectively,

$$\mathbf{B} = s\begin{bmatrix} 2 & 0 \\ 1 & \sqrt{3} \end{bmatrix} \tag{3.30}$$

and

$$\mathbf{B} = s\begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{3.31}$$

where s is a scale factor to be determined[4]. The first is the well-known hexagonal grid (figure 3.2), also denoted the A_2 lattice, and the second is the D_5^* lattice. The best known lattices for quantization of infinite uniform pdfs in 2-5 dimensions are A_2, D_3^*, D_4^* and D_5^*, respectively. These lattices are employed in our experiments in chapter 6. In [14], lattices for quantization purposes are thoroughly studied.

3.4 Truncation and scaling

As described previously in this chapter, a lattice quantizer is the intersection between a lattice Λ and a shape $\mathcal{S}$. The procedure to reject lattice points outside the shape, called *truncation* of the lattice, is of major importance for the performance of the resulting lattice quantizer.

Truncation for known distributions: Jeong and Gibson [30] argue that in a good lattice VQ, the lattice should be truncated by a contour of constant probability density for the

[4]Lattices can of course also be rotated and translated, but for high rates and smooth pdfs, these operations have little influence of the performance of a lattice VQ.

considered source, and design lattice VQs for Gaussian and Laplacian data. For the Laplacian pdf, this leads to truncation by a d-octahedron, which, mostly in combination with the integer lattice $\mathbb{Z}^d$, has received much attention since Fischer introduced the structure (Pyramid VQ) in the mid-80's. A recent reference on this topic is [31]; see also Swazek [32]. For a Gaussian pdf, the iso-probability contours are ellipsoids, and a corresponding truncating shape $\mathcal{S}$ is described by

$$\mathcal{S} = \left\{ \mathbf{x} \in \mathbb{R}^d : \mathbf{x}^{\mathrm{T}} \mathbf{C}_{\mathbf{x}}^{-1} \mathbf{x} < a^2 \right\} \tag{3.32}$$

where $\mathbf{C}_{\mathbf{x}}$ is the covariance matrix of the Gaussian input distribution, and a is a constant, determining the size of the ellipsoid. To truncate a lattice to the correct number of VQ points, the radius a above must be determined. An approximate value of a can be found by using the volume of the lattice Voronoi region, and for certain rates, a can be found by use of the theta function of the lattice.

A problem that may occur when lattices are truncated to a desired number of points is that a lattice normally has many points lying on the same distance from the origin (shell), and the truncation procedure may be required choose a few among those. To prevent lattice points to fall on the boundary, an arbitrary vector $\mathbf{v} \in \mathbb{R}^d$ can be added to the shape prior to the truncation:

$$C = \Lambda \cap (\mathcal{S} + \mathbf{v}) \quad \text{(or, equivalently, } C = (\Lambda - \mathbf{v}) \cap \mathcal{S}\text{)}. \tag{3.33}$$

After the truncation, the truncated lattice is moved to make the mean of all codevectors equal to the mean of the source. The choice of $\mathbf{v}$ can affect the performance of the resulting quantizer. We have experimented with four different methods to select $\mathbf{v}$:

I $\mathbf{v}$ is set to zero.

II $\mathbf{v}$ is selected as a very small (small compared to the basis vectors of the lattice) stochastic vector.

III $\mathbf{v}$ is selected as a stochastic vector with length in parity with the basis vectors of the lattice.

IV $\mathbf{v}$ is selected to minimize the energy of the resulting quantizer C,

$$\mathbf{v} = \underset{\mathbf{u} \in \mathbb{R}^d}{\operatorname{argmin}} \left\| \Lambda \cap (\mathcal{S} + \mathbf{u}) \right\|^2 \quad \text{where} \quad \left\| \mathcal{C} \right\|^2 = \sum_{k=1}^{N} \left\| \mathbf{c}_k \right\|^2 . \tag{3.34}$$

Method I leads to truncations that are natural for the chosen lattice, truncations were the outmost shell is full. This can of course only be achieved for certain values of the number of VQ points. Method II, III and IV can give arbitrary VQ sizes. Method IV has been used by Conway and Sloane [33] in a different application, and they also propose an iterative algorithm to perform the energy minimization. The first and second method (I and II) have proved best in the cases tested in this study. Since only a limited set of rates can be achieved

with method I, method II is preferred in this paper, although some results with method I are also reported.

After the truncation, the lattice VQ should be scaled to give the best possible performance. The scale factor can be approximated by use of high rate theory (see section 3.2), but to get better results an iterative procedure is often necessary, were the optimal scaling is found for a training database. Several authors have previously studied lattice scaling by iterative procedures, e.g., [8, 30, 34]. In [30], lattice VQ of iid Gaussian and Laplacian is treated, and the scaling is done by numerical optimization.

Data-optimized truncation: In applications, the source pdf is generally not analytically known, but described by an empirically collected database. In this case, we propose a data-optimized truncation, where every vector in the database is classified to its closest point in the full lattice, and the most probable lattice points are kept in the lattice quantizer. In contrast to truncation for known distributions, there is no way to avoid storing the truncation information for the data-optimized truncation. The algorithm is described in the following steps:

Step 1: An approximate scaling of the chosen lattice must be found. For iid Gaussian pdfs, and for pdfs that can be approximated as iid Gaussian, the high-rate scaling formulas in section 3.2 can be used. For unknown pdfs, ad-hoc scaling may be necessary. We have used a scaling rule that makes the granular distortion of the lattice equal to the distortion of a pdf-optimized quantizer with the desired rate, according to the Gaussian lower bound D_{GLB} (2.13) in section 2.3:

$$s = \sqrt{\frac{D_{\mathrm{GLB}}}{G}}, \tag{3.35}$$

where G is the quantization constant of the lattice. The estimated scale factor is only an approximation of the optimum scale, but the truncation procedure is not very sensitive to the scale, and mismatches are easily detected in step 3 of this algorithm. In all tested cases, this method has proven sufficient.

Step 2: Classify each vector in the database to the nearest lattice vector, by use of a nearest-neighbor algorithm for the chosen lattice [9]. The lattice points with the N highest probabilities become codevectors in the lattice quantizer.

Step 3: An optimal scale factor s^* for the lattice quantizer is found, by some numerical optimization method. If the scale factor is very different from the one found in step 1, go to step 2 and repeat the procedure using the new scale factor s^*.

Index-optimized truncation: In [33], Conway and Sloane introduce *Voronoi codes*, where the truncation is chosen as an integer multiple of the Voronoi region of the lattice. Forney subsequently generalizes the concept to other truncation shapes in [35]. With the

Voronoi codes, the indexing of the lattice VQ is greatly simplified. However, the Voronoi code truncation is generally not optimized for the pdf, and performance loss may result[5].

3.5 Indexing

In addition to the choice of Λ and $\mathcal{S}$, lattice VQ design involves one more issue; assignment of indices to the codevectors. This enumeration can be made aiming at several, partly conflicting, goals: *(i)* Memory saving. The indexing should have a mathematical formulation that is more compact than a full table. *(ii)* Fast encoding. The indexing should, in combination with one of the search algorithms that have been developed for lattices [9], yield a fast encoder $\mathcal{E}$. *(iii)* Fast decoding. The codevector should be rapidly retrievable from the index in the decoder $\mathcal{D}$. *(iv)* Symmetry. Characteristic for a lattice is that all points are alike in relation to the surrounding points. The indexing should preserve this property. In chapter 5, where an adjacency table is needed, the symmetry solves the memory problem. *(v)* Robustness. If the codebook is used for a noisy channel, bit errors should cause as little distortion as possible.

There exists an elegant solution of the indexing problem for Voronoi codes [33] in such a way that differences in indices reflect the relative position between codevectors. The method, based on modular arithmetics, satisfies *(i)–(iv)* above. On the other hand, Voronoi codes can only attain certain rates R, namely, those for which 2^R is an integer.

For a Gaussian probability density function, or other densities with rotational symmetry, it is beneficial if the truncation shape is as spherical as possible. Unfortunately, the d-sphere does not, in general, possess any of the appealing properties mentioned above. To combine a shape that is suitable for the source (such as the d-sphere for Gaussian data) with one that has a nice indexing (such as a Voronoi region), the former can be inscribed into the latter. This approach amounts to designing a larger set that includes the codebook, enumerating this larger set, and then disregarding the points that do not belong to the codebook. For this method, *(ii)-(iv)* above are satisfied. The larger set can for instance be chosen as a Voronoi code [33]. An alternative larger set is $\mathbf{B}^{\mathrm{T}} \cdot \mathbf{z}$, where $\mathbf{z}$ is a rectangular subset of the d-dimensional cubic lattice. Figure 3.5 illustrates the latter method for a 2-dimensional example, where a 19-point lattice VQ is enumerated by using a 25-point set, for which *(ii)-(iv)* are satisfied.

[5] Eyuboğlu and Forney shows in [29] that the performance loss is small for large dimensions.

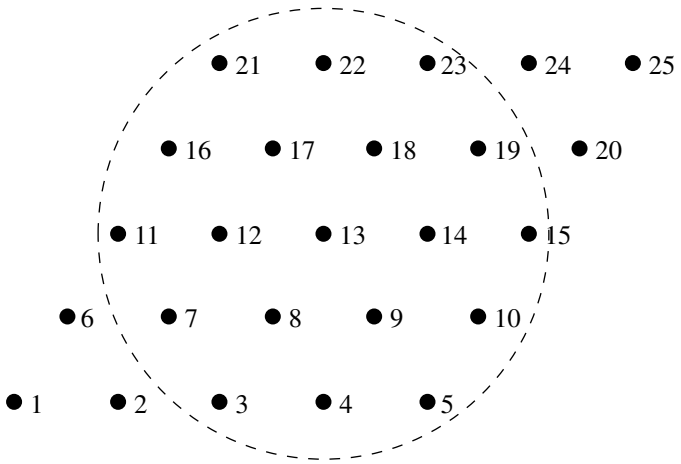


Figure 3.5. A 19-point lattice VQ, enumerated by using a 25-point set.

In the VQ design algorithm in chapter 4 and 5, we employ an indexing method in this category.

3.6 Lattice VQ examples

In figure 3.6, a lattice VQ and a pdf-optimized VQ are depicted. The SNR values for the lattice quantizer and the optimized quantizer are 14.6 dB and 15.3 dB, respectively.

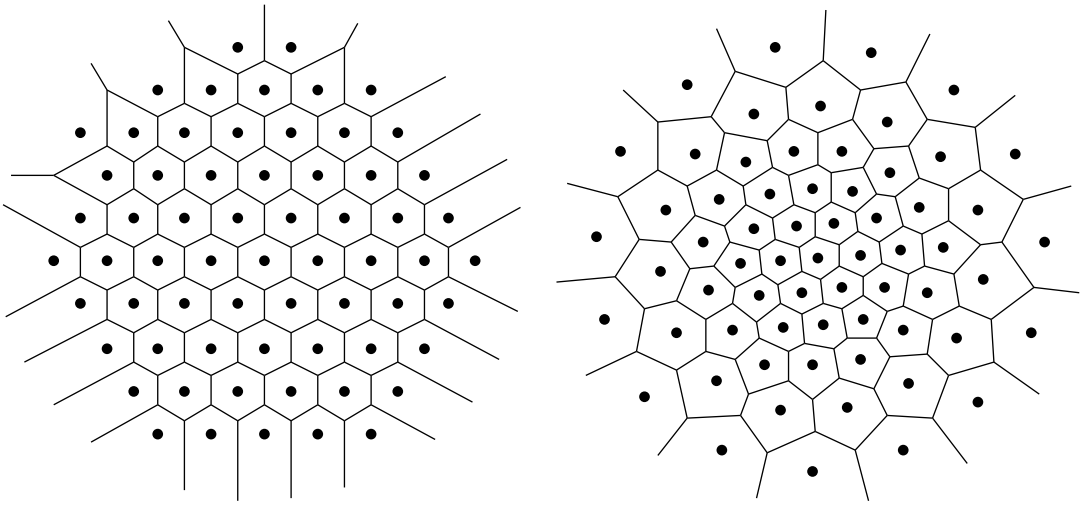


Figure 3.6. Two 64-point quantizers for a Gaussian pdf. Left: a lattice VQ. Right: a well-trained VQ.

In figure 3.7, the performance of lattice VQ is compared to pdf-optimized VQ for a 2- and a 5-dimensional iid Gaussian pdf.

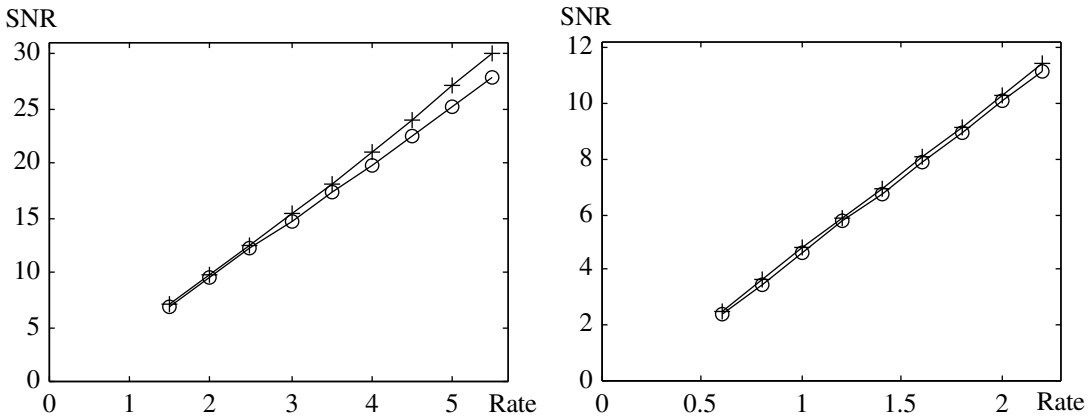


Figure 3.7. SNR as a function of rate for lattice VQ (○) and pdf-optimized VQ (+). Left: 2-dimensional VQ. Right: 5-dimensional VQ.

As predicted by the lattice high rate theory, the discrepancy between lattice VQ and pdf-optimized VQ increases for higher rates. More results for lattice quantization of Gaussian variables in 2 to 5 dimensions are reported on in section 6.2.

If the pdf-trained VQ in figure 3.6 is studied in detail, a feature of high rate quantizers can be observed: the structure is well-ordered, and the environment of the VQ points is locally similar to a lattice, at least for the points close to the center. This feature is exploited in the next chapter, to design VQs for fast search.

4. LATTICE-ATTRACTED VQ DESIGN

In this chapter, we propose an extension to standard VQ design algorithms, a *lattice-attracted* design algorithm, where the codebook is initialized with a truncated lattice, and the codevectors are updated to maintain a local lattice similarity for each iteration. The goal with this procedure is to make it possible to exploit the local lattice-similarity for fast nearest-neighbor search.

A sketch of a lattice-attracted algorithm is described in the following steps:

I: Initialize the VQ with a truncated lattice. An adjacency table for the lattice is also required, denoted the *lattice adjacency table*. This table consists of all neighbors to codevector **0** (vector zero), together with rules to compute the neighbors to an arbitrary point in the lattice.

II: Train the VQ with a conventional design method, but add procedures to approximately keep the initial set of neighbors, as defined by the lattice adjacency table.

The initialization procedure is described in section 4.1. In sections 4.2 and 4.3, we study how to extend two standard design algorithms, the generalized Lloyd algorithm [36] and a competitive learning algorithm [37], to approximately keep a predefined neighbor structure. In chapter 5, a novel lattice-based nearest-neighbor search method is described, based on the local lattice-similarity of the VQs trained with the proposed lattice-attracted algorithm. It is even possible to apply the fast search method during the training, as described in section 5.2.

The algorithm introduced here can, together with the specialized fast nearest-neighbor search method described in chapter 5, be viewed as a link between lattice quantization and unconstrained quantization, with the goal to combine some of the advantages of both methods.

4.1 Lattice initialization

Most iterative VQ design algorithms, such as the generalized Lloyd algorithm [36][6], or the competitive learning algorithm [37], can easily be trapped in a local distortion minimum when seeking the global minimum. A well-chosen initialization procedure can help the

[6]The generalized Lloyd algorithm is a direct generalization of a work by Lloyd, first presented in an unpublished technical note, “Least squares quantization in PCM”, at Bell Labs 1957.

algorithm to avoid local minima far from the global minimum. For example, the generalized Lloyd algorithm is often initialized by a splitting procedure, proposed by Linde et al [3] (the LBG algorithm). Another possibility is to initialize the VQ with a truncated lattice. Here, we use the lattice as a good initialization for further training, but also to find a lattice adjacency table for use in the fast search procedures described later.

The lattice initialization procedure starts with selection of a lattice with a good quantization constant G, as discussed in section 3.3. The lattice is truncated by any of the methods described in section 3.4. If the pdf of the source process is given by a database, the data-optimized truncation procedure can be used. For known pdfs, the lattice can be truncated by an iso-probability contour.

Now an adjacency table must be found for the chosen lattice. Voronoi neighbors of some standard lattices can be found in [9]. As discussed in section 3.1, the neighbors to a codevector can be computed by translation of the neighbors to any other codevector, so only neighbors to the zero codevector have to be stored. A simple enumeration technique is discussed in section 3.5, where the lattice VQ is enumerated by using a larger set with desirable properties. A possible larger set is given by $\mathbf{B}^T \cdot \mathbf{z}$, where $\mathbf{z}$ is a rectangular subset of the cubic lattice. The technique is illustrated in figure 3.5, where we see that the neighbors to an arbitrary point in the lattice VQ can be found by adding an offset of ± 1, ± 4 or ± 5 to the index of the point. This is not the most efficient method in terms of required storage, but it works and it is simple. A more storage-efficient larger set is the Voronoi codes discussed in section 3.5 and [33, 35], and these have been used in table 6.7. With the larger-set methods above, the neighbors to the actual codevector are found by a simple procedure; the index of the codevector is found, the offset to the wanted neighbor is added, and the codevector corresponding to the neighbor index is found[7]. The first operation, finding the index of a codevector, can be solved by storing a table of indices, with one integer index for each codevector. Adding offset is trivial, and finding the codevector corresponding to the neighbor index is either solved by looking in the index table, or in another table with index-to-codevector translations (or by a compromise between those two alternatives). See section 6.4 for storage requirements of the translation tables, and overhead complexity of the translation.

An alternative to ellipsoid truncation and larger-set indexing by table look-up, is direct use of the Voronoi codes in [33], for which no translation tables are necessary. However, a Voronoi-shaped truncation region is in general not optimal for the source pdf, and performance loss results.

For a complete description of the lattice Voronoi region, the distances to the neighbors are also stored. The set of neighbors to the zero codevector, together with the corresponding distances, describes the Voronoi region of any point in the lattice.

[7]Some of the codewords will not have a full set of neighbors, due to the truncation of the lattice. Missing neighbors are easily detected with the table look-up methods used here.

The features of the lattice initialization procedure are here illustrated by examples of two-dimensional vector quantizers. In figure 4.1, two 64-point VQs are plotted, directly after being initialized with a truncated lattice. Each VQ point and its neighbors, according to the lattice adjacency table, are connected by lines. The regular structure of the lattice initialization is clearly visible.

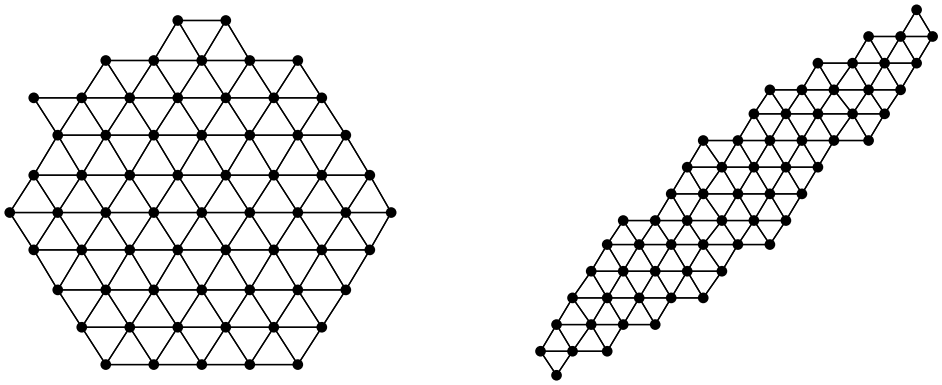


Figure 4.1. Neighbor structure (lines) for two lattice VQs (dots). Left: A lattice VQ optimized for uncorrelated Gaussian data. Right: A lattice VQ optimized for correlated Gaussian data, $\rho = 0.9$.

In the following sections, we will try to optimize the quantizers for the given source, while still maintaining a locally lattice-similar structure. The neighbors according to the lattice adjacency table, denoted the *lattice neighbors*, will deviate from the true Voronoi neighbors of the quantizer, but large similarities will remain, if the optimization procedure is successful.

4.2 Lattice attraction for the generalized Lloyd algorithm

The generalized Lloyd algorithm (GLA) is often used for unconstrained VQ design. In GLA, the two necessary conditions, (2.6) and (2.7), are alternatingly iterated until the quantizer has converged. GLA is a greedy algorithm, with the feature that the average distortion decreases for each iteration. This means that GLA finds the nearest local minima, and stops the iteration. To overcome this behavior, many methods have been proposed on how to add randomness to GLA [38], in order to make it possible to evade local minima. A good initialization is of prime importance for the success of GLA.

GLA is briefly described in table 4.1, step 1-3 and 5. To extend GLA to maintain the neighborhood structure as given by the lattice adjacency table, we add an extra step (step 4 in table 4.1), where all codevectors are moved a small step to increase the local lattice-similarity. This extra step can be implemented in several ways, and we describe one such way below. In advance, the codebook is initialized with a truncated lattice, and a lattice adjacency table is found, as described in section 4.1. After the standard GLA iteration, each codevector is moved a short step towards the centroid of its neighbors, according to the

distance to the corresponding neighbors in the lattice we want to mimic. In this way, the geometrical environment to each point in the VQ becomes more similar to the lattice, but each point has still a high degree of freedom during the training. The algorithm, from now on denoted *lattice-attracted GLA* or LA-GLA, is described in table 4.1, where step 4 is added to a standard GLA. In this algorithm description, the function to compute the lattice neighbors is denoted $\mathcal{N}(k,i)$, giving neighbor *k* of codeword *i* in the codebook. With l_k, we refer to the distance to neighbor *k* in the chosen lattice.

Table 4.1. The lattice-attracted GLA algorithm.

Step 1. Initialize the codebook $\mathcal{C}_1 = \left\{\mathbf{c}_1^{(1)}, \mathbf{c}_2^{(1)}, ..., \mathbf{c}_N^{(1)}\right\}$. Set $m = 1$.

Step 2. For the given codebook $\mathcal{C}_m$, classify each vector $\mathbf{x}$ in the training database $\mathcal{T}$ to a region $\Psi_k^{(m)}$, using the nearest neighbor partition

$$\Psi_k^{(m)} = \left\{\mathbf{x} \in \mathcal{T} : \left\|\mathbf{x} - \mathbf{c}_k^{(m)}\right\|^2 \le \left\|\mathbf{x} - \mathbf{c}_i^{(m)}\right\|^2 \text{ for all } i \in (1, N)\right\}$$

If a tie occurs, that is, if $\left\|\mathbf{x} - \mathbf{c}_k^{(m)}\right\|^2 = \left\|\mathbf{x} - \mathbf{c}_i^{(m)}\right\|^2$ for one or more *i*, assign $\mathbf{x}$ to the region $\Psi_i^{(m)}$ for which *i* is smallest.

Step 3. Compute a new codebook using the centroid condition

$$\mathbf{c}_k^{(m)} := \frac{1}{\left|\Psi_k^{(m)}\right|} \sum_{i=1}^{\left|\Psi_k^{(m)}\right|} \mathbf{x}_i$$

where the sum is over all training vectors $\mathbf{x}$ classified to $\Psi_k^{(m)}$, and $\left|\Psi_k^{(m)}\right|$ is the cardinality of the set $\Psi_k^{(m)}$ (the number of elements in $\Psi_k^{(m)}$). If $\left|\Psi_k^{(m)}\right| = 0$ for some *k*, use some other code vector assignment for that cell.

Step 4. Move all codevectors a small step ε_m to increase the lattice similarity,

$$\mathbf{c}_i^{(m+1)} = \mathbf{c}_i^{(m)} + \varepsilon_m \cdot \sum_{k=1}^{K(i)} \left(1 - \frac{w\left(\mathcal{N}(k,i)\right)}{\left\|\mathbf{c}_{\mathcal{N}(k,i)}^{(m)} - \mathbf{c}_i^{(m)}\right\| / l_k}\right) \cdot \left(\mathbf{c}_{\mathcal{N}(k,i)}^{(m)} - \mathbf{c}_i^{(m)}\right) \qquad i = 1, ..., N,$$

where $\mathcal{N}(k,i)$ is the lattice adjacency function, $K(i)$ is the number of neighbors to codeword *i*, and $w(j)$ is the average weighted distance between a codevector *k* and its neighbors,

$$w(j) = \frac{1}{K(j)} \cdot \sum_{k=1}^{K(j)} \left\|\mathbf{c}_{\mathcal{N}(k,j)}^{(m)} - \mathbf{c}_j^{(m)}\right\| / l_k \,.$$

The new set of vectors defines a new codebook, $\mathcal{C}_{m+1} = \left\{\mathbf{c}_1^{(m+1)}, \mathbf{c}_2^{(m+1)}, ..., \mathbf{c}_N^{(m+1)}\right\}$.

Step 5. Stop the iteration if some stopping criterion has been reached, for example if the average distortion for $\mathcal{C}_{m+1}$ has changed by a small enough amount compared to the distortion of $\mathcal{C}_m$. Otherwise, set $m := m + 1$ and go to step 2.

The step size parameter ε_m can be chosen to be constant over the training phase, or it can be a function of time. We have experimented with a linearly decreasing (to zero) step size,

$$\varepsilon_m = \varepsilon_0 \cdot \left(1 - \frac{m}{M}\right), \tag{4.1}$$

where ε_0 is the start step size and M is the total number of iterations of the algorithm. This choice makes the lattice attraction weaker and weaker, and at the end there is no attraction at all. We have experimented with different initial step sizes, and found that a value of ε_0 in the interval $0.05 - 0.1$ leads to good performance. The extra step is performed only once per iteration of the full training database, and thus the extra complexity is small.

In figure 4.2, two 64-point quantizers are depicted after being trained for a jointly Gaussian distribution with the LA-GLA algorithm, where the codebooks were initialized as in figure 4.1. We see that most of the lattice neighbor structure is retained, but that the quantizers are more optimized for the Gaussian pdf now.

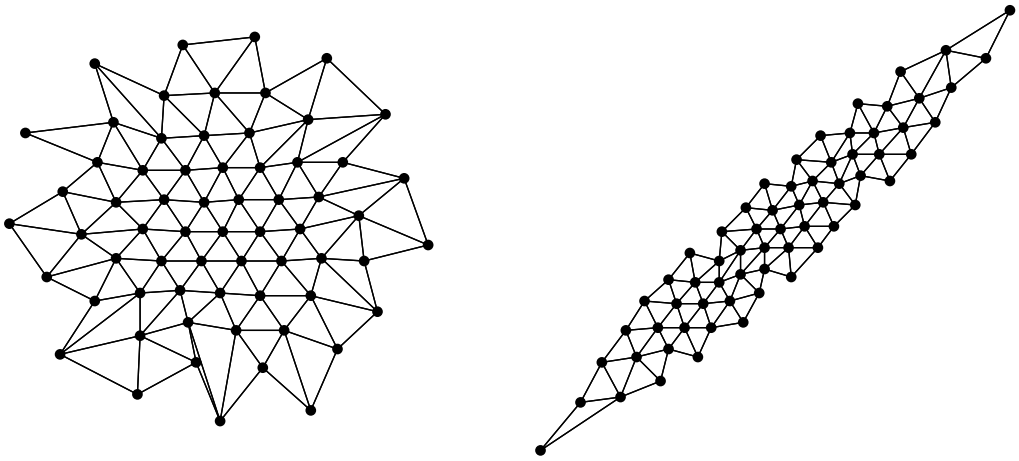


Figure 4.2. Two VQs optimized for Gaussian data, trained with the LA-GLA algorithm. Lattice neighbors are depicted as lines, and codevectors as dots. Left: uncorrelated data. Right: correlated Gaussian data, $\rho = 0.9$.

Results from simulations with the LA-GLA method are reported on in section 6.3.

4.3 Competitive learning with lattice attraction

Competitive learning (CL) [37] was first developed for training of artificial neural networks, but can also be used for vector quantization training. In the CL algorithms, the training vectors are presented one by one, and only one codevector (the closest one) is adjusted for each input vector. The learning rule of CL can be derived from the two necessary conditions in section 2.2 [39], which make CL and GLA essentially equivalent. The main difference is that GLA works in a batch mode, were all training vectors are presented before the codevectors are adapted, as opposed to the sample iterative technique used in CL

algorithms. Another important difference is that in contrast to GLA, the CL algorithm is not greedy; the average distortion does not necessarily decrease at each iteration. This allows the CL algorithm to evade some local minima.

In [37], Kohonen presents the *self-organizing feature map*, which extends CL by modifying not only the winner at each iteration, but also neighbors to the winner according to some topological map. The map is often a two-dimensional square lattice, where the neighbors can be easily computed. A feature of Kohonen training is that the structure of the map is imposed on the quantizer. Knagenhjelm [40] uses a *Hamming map*, in order to train VQs where the Hamming distance between codewords and the Euclidean distance between codevectors are closely related. This is shown to substantially robustify the VQ for transmission over a noisy binary symmetric channel.

The self-organizing feature map is a straightforward way to attract the quantizer to the lattice. The neighbors in the map are given by the lattice adjacency table, and the winning candidate is modified together with all neighbors in the table for each presentation of input data. The algorithm is described in table 4.2.

Table 4.2. The competitive learning algorithm with a lattice topology map.

Step 1. Initialize the codebook $\mathcal{C}_1 = \{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_N\}$. Set $m = 1$.

Step 2. A random vector $\mathbf{x}_m$ is drawn from the training database. For the input data $\mathbf{x}_m$, find the winning candidate according to the quadratic error criterion,

$$\mathbf{c}^* = \underset{\mathbf{c} \in \mathcal{C}_m}{\operatorname{argmin}} \|\mathbf{x}_m - \mathbf{c}\|^2.$$

Step 3. Modify the winning codevector as

$$\mathbf{c}^* := \mathbf{c}^* + \eta_m \cdot (\mathbf{x}_m - \mathbf{c}^*).$$

where the "temperature" η_m is linearly decreasing from an initial temperature η_0:

$$\eta_m = \eta_0 \left(1 - \frac{m}{M}\right).$$

Step 4. Modify the neighbors to the winning candidate a small step ε_m, according to

$$\mathbf{c}_k := \mathbf{c}_k + \eta_m \cdot \varepsilon_m \cdot (\mathbf{x}_m - \mathbf{c}_k) \ , k = 1, ..., K.$$

where $\mathbf{c}_k$ is one of the totally K neighbors (found in the lattice adjacency table) to $\mathbf{c}^*$.

Step 5. If $m = M$, then stop the iteration. Otherwise, set $m := m + 1$ and go to step 2.

The neighbor step size ε_m is, as in the LA-GLA, linearly decreasing,

$$\varepsilon_m = \varepsilon_0 \left(1 - \frac{m}{M}\right). \qquad (4.2)$$

The resulting CL algorithm is denoted the *lattice-attracted competitive learning* (LA-CL) algorithm. Results of simulations with this algorithm are presented in chapter 6.

5. FAST SEARCH OF LATTICE-ATTRACTED VQ

In [11], an algorithm for fast search of arbitrary VQs is described. With this algorithm, denoted the *steepest neighbor descent* (SND) algorithm, an adjacency table is precomputed, consisting of all Voronoi neighbors to all codevectors in the VQ (how to find the adjacency table is described in [11]). When the table is found and stored, the actual quantization can begin. For each input vector $\mathbf{x}$, one of the codevectors in the codebook is selected as a starting hypothesis $\mathbf{c}^{(0)}$. The distance between $\mathbf{x}$ and $\mathbf{c}^{(0)}$ is computed, and then the distances between $\mathbf{x}$ and the neighbors to $\mathbf{c}^{(0)}$ (found in the adjacency table) are computed. When all neighbor distances have been computed, the neighbor closest to $\mathbf{x}$ becomes the new hypothesis $\mathbf{c}^{(1)}$.

This procedure is repeated until a hypothesis vector is found whose neighbors are all worse. It can easily be shown that when a codevector with lower distance to the input vector than all its neighbors is found, this vector is the optimal codevector (see (2.10)).

The main disadvantage of the SND algorithm is the storage requirements for the pre-computed adjacency table, typically many times the required storage of the codebook. For example, a 12 bit 6-dimensional VQ requires around 700 kbyte storage for the adjacency table [11], and this is impractical for many applications.

Lattices have a feature that can be exploited to reduce the storage requirements for the SND algorithm; all neighbors to an arbitrary point in a lattice can be found by translation of the neighbors to the zero lattice point. To find the neighbors to an arbitrary point in a lattice VQ, the neighbors to the zero point are translated, and the set of neighbors is truncated by the global truncation rules. Thus, we can apply the SND algorithm to a lattice VQ, supported only by the neighbors to a single region. However, this would not be a very competitive algorithm, since fast specialized search algorithms have been developed for many important lattices [33]. A better choice is to apply the low-storage SND algorithm to the well-performing lattice-attracted quantizers from chapter 4. These quantizers are trained to maintain a lattice neighbor structure, and are well suited for low-storage SND search.

In this chapter, we discuss how to apply the steepest neighbor descent method to the quantizers trained by LA-GLA or LA-CL algorithm.

5.1 An extended SND algorithm

Here, we will propose an SND algorithm to suit the lattice-attracted quantizers from chapter 4. The lattice neighbors of the lattice-attracted quantizers (c.f. figures 4.1 and 4.2) are not always in perfect correspondence with the real Voronoi neighbors. False neighbors, i.e., codevectors listed as lattice neighbors without being Voronoi neighbors, constitute no problem, but not listed Voronoi neighbors can lead to erroneous decisions, and must be considered.

An important issue is the starting point of the algorithm, i.e., the choice of an initial hypothesis codevector. For the tested Gaussian densities, the trained lattice-attracted quantizers show a high degree of similarity with the lattice quantizer used for the initialization of the LA-GLA and LA-CL algorithms; the codevectors stay in general fairly close to their initial positions. Thus, a good starting hypothesis is the vector found by nearest-neighbor search of the initial lattice quantizer. For many important lattices, nearest neighbor search can be done with very low complexity [9]. No extra storage is required for this, just a search algorithm for the chosen lattice.

We have extended the SND algorithm to handle the special problems with an incomplete adjacency table, and also to exploit the lattice-similarity to find a good starting point. Three extensions have been used:

I An initial hypothesis is found by nearest-neighbor search of the chosen lattice.

II If the current hypothesis codevector is closer to the input vector than all of its neighbors, the neighbor descent search continues from the second best vector. This procedure is repeated until no improvement is obtained.

III When the SND terminates and declares a winning codeword, an *exception table* is consulted, including Voronoi neighbors not found in the lattice adjacency table. If the winning codeword is found in the exception table, the listed extra neighbor(s) is also tested.

The exception table should be constructed prior to the actual quantization. All the missing Voronoi neighbors do not have to be included in the exception table, only those that lead to a substantially higher distortion if not included. The exception table can be found by running a full search in parallel with the SND search for a training database, and observing when the answers from the two search procedures differ.

The first extension requires a lattice nearest-neighbor search prior to the VQ search. The complexity of this extension varies with the effectiveness of the search algorithms for the actual lattice, but for the lattices used here, the complexity corresponds to 0.5-2 extra distance computations. No extra storage is needed. The second extension has experimentally shown to lead to a few additional distance computations for each input vector, compared to the standard SND algorithm, but no extra storage is required. The third extension, the

exception table, requires some extra storage, but the extra search complexity is small, since the exception table is seldom consulted.

Experiments show that if the performance loss compared to a full search is required to be less than 0.01 dB, the exception table can be very small, typically a few entries for the 2-dimensional VQs tested here, and 20-30 entries for the high rate 5-dimensional VQs. If no performance loss at all is allowed, the 5-dimensional VQs may require an exception table that includes up to 10-15% of the vectors in the codebook, to compensate for all missing neighbors, even though these occur with a probability close to zero.

If the exception tables are excluded, some performance loss is inevitable. The 5-dimensional VQs require larger exception tables to reach 0.01 dB performance loss than the 2-dimensional VQs, but on the other hand, if the exception tables are excluded, the performance loss of the 5-dimensional VQs is small, for the tested VQs always less than 0.05 dB. In section 6.4, we report the performance, in terms of storage and search complexity, for quantizers where the exception table is designed for “almost lossless” (less than 0.01 dB loss) operation.

The extended SND algorithm (eSND) is described in table 5.1.

Table 5.1. The extended steepest neighbor descent (eSND) algorithm.

Step 1: Find an initial hypothesis codevector $\mathbf{c}^*$, by a lattice nearest-neighbor search. Set the temporary codevector $\mathbf{c}$ to null.

Step 2: Find the lattice neighbors to $\mathbf{c}^*$, by look-up and translation of the lattice adjacency table.

Step 3: Compute the distortion of all untested neighbors. If a better codevector than $\mathbf{c}^*$ is found, this becomes the new hypothesis $\mathbf{c}^*$, and the execution continues at step 2. If no better neighbor can be found, continue to step 4.

Step 4: If the current hypothesis $\mathbf{c}^*$ is equal to the temporary codevector $\mathbf{c}$, continue to step 5. Otherwise, set the temporary codevector $\mathbf{c}$ to the second best codevector found up to then, set $\mathbf{c}^* = \mathbf{c}$, and go back to step 2.

Step 5: If the current best hypothesis is listed in the exception table, compute the distortion of the extra neighbor(s) as given by the exception table.

Step 6: The best codevector found until now is returned.

The algorithm works well for Gaussian data. An interesting question is how well it generalizes to other pdfs. The simple answer is that it generalizes to pdfs that can be well quantized using a quantizer with locally lattice-similar structure. These include pdfs where direct lattice quantization works well, and thus the VQ points typically move only a small distance from the lattice initialization. It also generalizes to pdfs for which a multidimensional compander in combination with a lattice quantizer works well (see, e.g., [41] for a treatment

of this subject). However, the question if the algorithm works well for arbitrary pdfs is a subject for further research.

In section 6.4 we report on the search complexity reduction that can be achieved with the eSND algorithm. In section 5.2, we study how to apply the eSND algorithm already during the design phase, with a design complexity reduction as result.

5.2 Fast search during the design phase

To speed up the design procedure by the LA-GLA and LA-CL algorithms, the fast search procedure can be incorporated in the training. The introduction of the eSND search during the design phase leads to a few problems. First, the exception table in the eSND algorithm must be constructed ”on-line” during the design process. The exception table during design may be far from complete; the training has experimentally shown to be fairly insensitive to a few misclassifications. We have experimented with construction of an exception table after the first iteration of the GLA algorithm, by doing a full search in parallel with the eSND. For the following iterations only eSND search is performed. After some iterations, it might be necessary to reconstruct the exception table.

Another problem we encountered in the development of the LA-CL method was a break-down tendency (failure to improve the VQ) for high initial temperatures η_0. This is caused by the random reordering of codevectors that occur for high temperatures, destroying the well-ordered initial lattice structure. When the lattice structure is destroyed, the eSND search fails more often to find the optimal codevector, and as a result the VQ is adapted to destroy the lattice structure even more. However, the break-down temperature is distinct and well above realistic start temperatures, so the problem is easily avoided. The LA-GLA algorithm has not shown any tendencies to break down for the problems treated in this report.

5.3 Related work

In the literature, some other reports on fast search for unconstrained VQs can be found. As discussed earlier, there are some methods based on the neighbor descent concept. These algorithms show similar performance as the proposed eSND algorithm for lattice-attracted VQs, but the storage requirement for the adjacency table is typically many times the required storage of the codebook [10, 11]. In [42], only a fraction of the full adjacency table is stored, with a suboptimal search procedure as a result.

Another method is the *K-d tree* technique, proposed in [43], and further developed in, e.g., [13]. A binary tree, with hyperplane decision tests at each node, is precomputed and stored. The decision tree leads to one of a set of terminal nodes, where small sets of still eligible candidate vectors are listed.

In the *projection* technique [44], a rectangular partition of the space is precomputed and stored. During the search, the rectangular cell containing the input vector is found, and the

distances to a small number of eligible codevectors are computed. The number of distance calculations with this method is typically very small, but the overhead complexity is considerable.

Anchor point algorithms [12, 45] are algorithms where VQ points are excluded from the search by use of the triangle inequality. The distances from a small set of anchor points to each of the codevectors are precomputed and stored. The encoder then computes the distance between the input vector and each anchor point, and a large number of codevectors can be eliminated from the nearest neighbor search.

In [46], a Kohonen feature map is used as a basis for a fast search algorithm. However, the search algorithm shows poor performance, with a high percentage of misclassifications, due to the selection of a map that is not a good quantizer in itself.

For comparison, we have included measurements of an anchor point algorithm and the projection technique, in section 6.4.

6. EXPERIMENTS

In many real-world applications employing vector quantization, the Gaussian distribution is used as a model for the incoming data, and also as a model of the quantization error. This is mainly because it is possible to theoretically compute important parameters for Gaussian pdfs, but also because the Gaussian distribution is often a good approximation to the pdf of the actual data. This makes the performance of quantization of Gaussian variables interesting.

In this chapter, we present simulation results of lattice quantization and lattice-attracted VQs, and study their performance for Gaussian pdfs. In section 6.1, we describe the databases used in the experiments. In section 6.2, the performance for lattice VQ of Gaussian data is given, and in section 6.3, the performance of the new lattice-attracted method is tabulated. The achievable search complexity reductions and extra memory requirements for the eSND method are given in section 6.4, where it is also compared to an anchor point algorithm.

6.1 Databases

All Gaussian variables are generated by the Box-Müller method, using a well-tested random number generator from [47]. Both correlated and uncorrelated databases are generated. The correlated data are sequences of samples, drawn from a first order Markov process with correlation coefficient $\rho = 0.9$.

6.2 Results for Gaussian variables

In this section, we present the performance of lattice quantization of Gauss-Markov processes. The lattices are truncated as described in section 3.4, with method II for known pdfs, and the optimal scale factors are determined by an iterative procedure, using a database of 200 000 samples. For comparison, we also present SNR values for optimized Gaussian vector quantization (20 million iterations of a CL algorithm are used to train the quantizers). For the performance evaluation, an independent evaluation database with 1 million Gaussian vectors is used, both for lattice VQs and pdf-optimized VQs.

In table 6.1, we present signal-to-noise-ratios (SNR) for quantization of an iid Gaussian pdf[8].

Table 6.1. SNR (in dB) for lattice VQ and pdf-optimized VQ (inside parenthesis), for quantization of uncorrelated Gaussian vectors.

Number of codewords	Dimension of VQ			
	$d=2$	$d=3$	$d=4$	$d=5$
8	6.78 (6.96)	4.29 (4.48)	3.16 (3.34)	2.38 (2.53)
16	9.48 (9.68)	6.20 (6.29)	4.41 (4.67)	3.48 (3.66)
32	12.09 (12.44)	7.91 (8.10)	5.90 (5.99)	4.59 (4.77)
64	14.64 (15.29)	9.68 (9.95)	7.17 (7.36)	5.76 (5.84)
128	17.22 (18.18)	11.48 (11.83)	8.54 (8.75)	6.77 (6.93)
256	19.85 (21.10)	13.24 (13.74)	9.90 (10.15)	7.89 (8.05)
512	22.47 (24.04)	14.97 (15.66)	11.22 (11.57)	8.98 (9.17)
1024	25.11 (27.03)	16.71 (17.62)	12.59 (13.00)	10.07 (10.31)
2048	27.75 (29.88)	18.45 (19.62)	13.91 (14.49)	11.12 (11.47)

We see that lattice quantization can give competitive performance for low and medium rates, but for higher rates, the pdf-optimized VQ is significantly better. As predicted by the high-rate lattice theory in section 3.2, a lattice quantizer is inferior to a pdf-optimized quantizer when the rate is high.

We also wanted to examine the importance of the truncation procedure. For this purpose, we have applied truncations that are natural for the chosen lattice, i.e., truncations that acknowledge the shell structure of the lattice, and keep the outmost shell fully populated (method I in section 3.4). This can of course only be achieved for certain number of points. For the D_5^* lattice, the number of points in the shells[9] is, from inside out, given by the theta series {1, 10, 32, 40, 80, 160, 90, 112, 320,...}, and thus the number of points in a quantizer with fully populated shells are {1, 11, 43, 83, 163, 323, 413, 525, 845, ...}. In figure 6.1, we compare the performance of lattice VQs with fully populated shells with VQs where the number of points is an integer power of 2.

[8]Note that the results for high-rate pdf-optimized quantizers show signs of undertraining; especially the SNR values for 2 dimensions, 2048 codewords could be improved with longer training.
[9]Other theta series are possible if the lattice is translated.

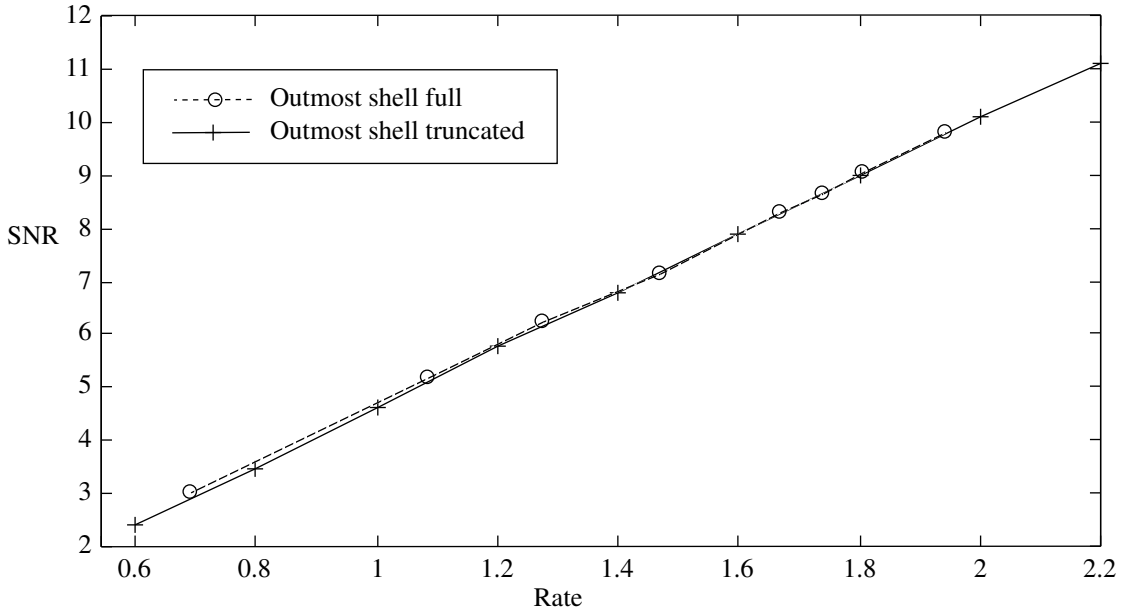


Figure 6.1. Performance for a truncated lattice VQ on a 5-dimensional iid Gaussian pdf. The crosses (x) indicate performance for lattice VQ where the number of points is truncated to an even power of two, and the circles (o) indicate the performance with a fully populated outmost shell.

We see that for low rates, the truncation where the outmost shell is fully populated has a performance advantage, but for higher rates the ”unstructured” truncation procedure gives equivalent performance.

In table 6.2, we present signal-to-noise-ratios for lattice quantization of a first order Gauss-Markov process with correlation coefficient 0.9.

Table 6.2. SNR (in dB) for lattice VQ and pdf-optimized VQ (inside parenthesis), for a first order Gauss-Markov process, with correlation coefficient 0.9.

Number of codevectors	**Dimension of VQ**			
	***d*=2**	***d*=3**	***d*=4**	***d*=5**
8	9.72 (10.83)	9.20 (9.37)	8.19 (8.48)	7.43 (8.09)
16	12.48 (13.55)	10.45 (11.41)	9.23 (10.20)	8.37 (9.39)
32	15.13 (16.25)	12.30 (13.21)	10.50 (11.66)	9.48 (10.69)
64	17.98 (19.05)	14.08 (15.01)	12.08 (13.03)	11.02 (11.85)
128	20.82 (21.87)	16.16 (16.85)	13.44 (14.40)	11.83 (12.96)
256	23.28 (24.81)	17.80 (18.71)	14.95 (15.77)	13.19 (14.05)
512	25.80 (27.72)	19.69 (20.60)	16.46 (17.16)	14.37 (15.14)
1024	28.63 (30.67)	21.36 (22.51)	17.69 (18.56)	15.54 (16.23)
2048	31.24 (32.82)	23.16 (24.39)	19.18 (19.97)	16.70 (17.35)

We see that for correlated Gaussian data, pdf-optimized vector quantizers have in most cases a significant performance advantage over lattice quantizers.

6.3 Lattice-attracted VQ design performance

With the new lattice-attracted VQ design methods, an interesting question is if the lattice attraction leads to loss of performance compared to unconstrained VQ training. To inves-

tigate this, the performance for quantizers trained until convergence with the different methods are compared in table 6.3. The SNR values are averaged over 20 simulations with different training databases (different seeds for the random number generator). The evaluation database consists of one million Gaussian vectors. Even though the GLA algorithm is normally aborted when the distortion change is small enough, we have here chosen to run all algorithms for a predetermined number of iterations (100 million iterations are performed in all cases, where one iteration consists of finding the closest codevector to an input vector). The chosen design time is large enough for all the methods reach convergence, i.e., the results do not improve for longer training. The size of the training database is limited (500 000 vectors) for the batch algorithms, LBG and LA-GLA, but for the competitive learning methods, the database size is "unlimited"; a new Gaussian vector is drawn for every iteration.

Table 6.3. SNR (dB) for quantizers trained until convergence with the different methods.

dim, size, corr	CL	LA-CL	LBG	LA-GLA
d=2, N=64, ρ=0	15.30	15.30	15.27	15.27
d=2, N=64, ρ=0.9	19.05	19.05	19.03	19.02
d=3, N=128, ρ=0	11.85	11.85	11.82	11.82
d=3, N=128, ρ=0.9	16.87	16.87	16.83	16.82
d=5, N=1024, ρ=0	10.32	10.32	10.25	10.26
d=5, N=1024, ρ=0.9	16.23	16.23	16.20	16.20

Note that the CL algorithms perform slightly better than LBG or LA-GLA. A reason for the inferiority of the GLA-based algorithms is the limited training database, making the greedy GLA-based algorithms more easily trapped in local minima. From the numbers in table 6.3, we conclude that the lattice attraction does not decrease the performance of the fully trained VQ, neither for GLA nor CL. For these extremely well-trained quantizers, the lattice-constraint is mainly a question of indexing of the codevectors; for all methods, the resulting structures of the quantizers are very similar. This indicates that an indexing procedure could be applied after the training procedure to make the fast eSND search possible. However, it would then be impossible to apply the eSND during the training.

In reality, it may be impractical with the tedious train-until-convergence used above, and the database size is also often limited. A more realistic database can have a size that is only 100 times the number of codewords, and in some cases even less. In figure 6.2, we compare the different design methods for limited design time and database size.

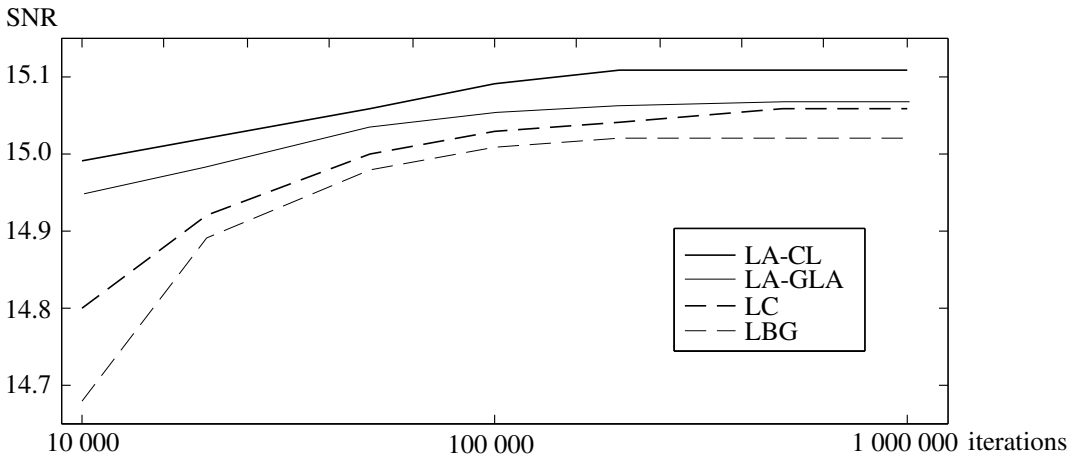


Figure 6.2. SNR as a function of number of iterations for design of a 64-point 2-dimensional VQ. For all methods, the training database contains 5000 vectors, drawn from an iid Gaussian pdf. The LBG algorithm uses a split initialization technique, while the other algorithms are initialized with a truncated lattice, giving an initial SNR of 14.6 dB.

We see that the lattice-attracted design methods reach a higher SNR for a limited database size, due to the attraction to a well-ordered lattice structure, a structure that otherwise can be hard to reach for limited training times and databases. No method reaches an SNR close to the optimum 15.3 dB (table 6.3).

The results in this section seem to indicate that the CL-based algorithms should be preferred for VQ design. However, the tuning of the starting temperature for the CL algorithms can be tedious, and the empty-cluster-problem is simpler to handle in GLA-based algorithms. Thus, LBG and LA-GLA may still be preferable in some applications.

6.4 eSND performance

In this section, we report on the performance of the eSND algorithm, in terms of search complexity and storage requirements. For comparison, we have also included measurements of an anchor point algorithm, using the same databases.

Search complexity: We have applied the eSND algorithm, described in chapter 5, to quantizers trained with LA-GLA. The exception tables are designed for “almost lossless” operation, with a performance loss compared to full search that is less than 0.01 dB. The average and maximum number of distance computations are listed in table 6.4 for iid Gaussian, and in table 6.5 for Gauss-Markov ($\rho = 0.9$). The number of distance computations of a full search is of course equal to the number of codewords in the quantizer.

Table 6.4. Average and maximum (within parenthesis) number of distance computations for the lattice-attracted quantizers. The database consists of uncorrelated Gaussian vectors.

Number of codewords	Dimension of VQ			
	$d=2$	$d=3$	$d=4$	$d=5$
8	5.8 (8)	6.1 (8)	7.7 (8)	6.7 (8)
16	7.7 (12)	10.4 (16)	12.2 (16)	12.0 (16)
32	9.1 (12)	13.8 (25)	18.7 (32)	22.1 (32)
64	9.9 (12)	17.3 (28)	24.7 (51)	32.7 (63)
128	10.6 (13)	19.6 (28)	30.9 (63)	42.1 (89)
256	10.6 (15)	21.5 (31)	36.1 (69)	53.2 (124)
512	10.5 (16)	23.3 (36)	41.5 (79)	64.8 (145)
1024	10.5 (16)	25.2 (40)	45.7 (91)	75.3 (167)
2048	10.5 (16)	25.2 (44)	50.2 (96)	81.8 (179)

Table 6.5. Average and maximum (within parenthesis) number of distance computations for the lattice-attracted quantizers. The database consists of correlated ($\rho = 0.9$) Gaussian vectors.

Number of codewords	Dimension of VQ			
	$d=2$	$d=3$	$d=4$	$d=5$
8	4.3 (6)	3.8 (5)	3.8 (5)	3.9 (5)
16	6.5 (9)	6.9 (11)	7.2 (11)	7.1 (11)
32	7.9 (11)	9.8 (16)	12.4 (22)	12.7 (24)
64	9.1 (12)	13.4 (24)	15.8 (30)	16.1 (30)
128	9.9 (13)	15.7 (25)	22.5 (44)	27.1 (58)
256	10.5 (14)	19.0 (30)	28.2 (52)	33.9 (70)
512	10.7 (15)	20.8 (32)	33.1 (64)	45.8 (100)
1024	10.8 (17)	23.2 (36)	39.8 (79)	56.0 (126)
2048	10.9 (19)	24.3 (40)	44.0 (84)	66.6 (148)

We see that a significant reduction of the number of distance computations is achieved for the eSND method, and also that the maximum is reasonable (measured for one million test vectors).

Besides of the distance computations, some additional overhead for the eSND algorithm is unavoidable. The initial hypothesis codevector is found by searching the closest vector in the lattice associated with the lattice-attracted VQ. This procedure is not very complex due to the regular structure of the lattice; for the lattices employed here, the procedure involves a rescaling of the input vector, adding an offset vector and rounding all elements towards the nearest integer. The total overhead complexity for finding the initial hypothesis is less than two extra distance computations for the lattices used here. More about lattice nearest-neighbor search algorithms can be found in [9]. There is also overhead for each distance computation. When a new hypothesis codevector is found, the lattice index of the codevector must be found, by table lookup as described in section 4.1. For each distance computation, an integer is added to the lattice index, and the codevector corresponding to the sum is found by table lookup[10]. The overhead depends on the efficiency of integer arithmetics of the given

[10]If a Voronoi code is used, the table lookups are unnecessary; the indices of the codewords are given by the sorting of the codebook. But Voronoi codes may lead to performance loss.

processor, but for the hardware used here (DEC Alpha), the overhead complexity is only a fraction of the complexity of the distance computations.

It is interesting to compare the eSND method with other fast nearest-neighbor search methods (see section 5.3). In comparison with other neighbor descent methods, eSND has a slight advantage, because of the good initial hypothesis given by the lattice search, but the overall performance should be similar due to the similar approaches. Among other methods, anchor point algorithms are well-known. We have implemented an anchor-point algorithm, IFAP-AESA [12]. IFAP-AESA substantially reduces the number of L2-norm distance computations, at the cost of a number of L1-norm distance computations. A procedure similar to the standard *partial distance* technique [44, 48] is employed for the L1-norm computations to further reduce the complexity. We have also implemented the projection method [44], briefly described in section 5.3. The rectangular partition is optimized for ”almost lossless” operation, with at most 0.01 dB performance loss compared to full search.

While the complexity of full search and eSND is essentially proportional to the number of L2-norm distance computations, this is not true for IFAP-AESA and the projection method. Therefore, we report the complexity in the average number of floating point multiplications, additions, comparisons and integer operations (given as a proportionality constant) per input vector. The additional overhead for eSND is described above, and for IFAP-AESA the overhead consists of frequent absolute value computations and table look-ups. The overhead complexity for the projection method is considerably higher than for the other methods, with a large number of integer operations. Actually, the complexity of the projection method is dominated by the integer operations for the cases tested here.

The nearest-neighbor algorithms are compared in table 6.6.

Table 6.6. Average number of multiplications, additions, comparisons and integer operations for a full search, for an anchor point algorithm, IFAP-AESA, the projection method and for the eSND algorithm. The database consists of uncorrelated Gaussian vectors.

Dimension d, VQ size N	**Multiplications, Additions, Comparisons (Integer operations)**			
	Full search	**IFAP-AESA**	**Projection**	**eSND**
$d=2$, $N=64$	128, 192, 63 ($\propto N \cdot d$)	11, a=168, 117 ($\propto a$)	3, 4, 15 ($\propto N \cdot d$)	20, a=30, 20 ($\propto a$)
$d=3$, $N=128$	384, 640, 127 ($\propto N \cdot d$)	23, a=556, 386 ($\propto a$)	6, 11, 26 ($\propto N \cdot d$)	59, a=98, 39 ($\propto a$)
$d=5$, $N=1024$	5120, 9216, 1023 ($\propto N \cdot d$)	70, a=8286, 5983 ($\propto a$)	26, 47, 60 ($\propto N \cdot d$)	377, a=678, 150 ($\propto a$)

The number of integer operations for the projection method and for full search is proportional to the VQ size N times the dimension, while the number of integer operations for eSND and IFAP-AESA is proportional to the number of distance computations (which is the sum of L1-norm and L2-norm distance computations for IFAP-AESA). This means that the number of integer operations for IFAP-AESA and eSND grows much slower than for full search and the projection method.

We see that IFAP-AESA radically reduces the number of multiplications, but that the number of additions and comparisons remains high. IFAP-AESA can only compete with the other algorithms for hardware where the multiplication cost is dominating, but in terms of FLOPS (floating-point operations per second), IFAP-AESA is inferior. On the other hand, the projection algorithm outperforms the other algorithms in terms of FLOPS. However, as discussed above, the overhead complexity for the projection method is considerably higher, and which of the two methods that is the fastest in practice is dependent on the efficiency of the hardware.

Storage requirements: To use the eSND fast search algorithm, we must precompute and store an adjacency table, an exception table, and tables to aid translation from codebook index to lattice index and vice versa. In table 6.7, the required storage of the tables and the codebook is given for a few VQ examples.

Table 6.7. Relative and absolute storage requirements (in bytes) for examples of iid Gaussian quantization. The codebooks are stored as 4-byte floating point numbers, and the tables consist of one- or two-byte integer values. The total storage is given in percentage of codebook only storage.

Storage requirements	$d=2,\ N=64$	$d=3,\ N=128$	$d=5,\ N=1024$
Codebook	$64\times2\times4$ = 512	$128\times3\times4$ = 1536	$1024\times5\times4$ = 20480
Adjacency table	6	14	$62\times2=124$
Exception table	0	3	$15\times2=30$
Translation tables	145	371	$4149\times2=8298$
Total storage	129%	125%	141%

As seen in the table, the storage requirements are dominated by the codebook and the translation tables. The larger extra storage of the 5-dimensional VQ depends on that 2 instead of 1 byte is required to encode the 1024 codewords. Since we only consider unconstrained VQs, the codebook size can not be reduced, unless the precision is somehow reduced. It is possible to reduce the storage requirements for the translation tables, at the cost of extra overhead time for the eSND search.

The anchor point algorithm requires storage of a floating point table with size $(d+1)/d$ times the size of the codebook. For the 2-, 3- and 5-dimensional cases above, the total storage, in percent of codebook only storage, are 250%, 233% and 220%, respectively.

For the projection method, a rectangular partition of the space, and a set of candidate codewords for each rectangular cell, are precomputed and stored. The total storage, in percent of codebook only storage, are 350%, 350% and 400% for the cases above.

7. SUMMARY

In this report, lattice-based quantization was studied, both from a theoretical and a practical viewpoint. Lattice-based quantization is a generalization of conventional lattice quantization, by allowing modifications of the regular lattice structure while still maintaining a local lattice-similarity.

For conventional lattice quantization, high rate theory was developed. The high rate theory leads to lattice VQ design rules, and to new insights in the performance of lattice quantization. An important conclusion was that for high rates, lattice quantization is severely inferior to optimal vector quantization. Practical solutions to problems in lattice quantization, such as truncation and scaling, were discussed, and the performance of lattice quantization of Gaussian variables was presented.

To overcome the inherent shortcomings of lattice quantization, we proposed a novel lattice-based technique for VQ design, with the feature that the resulting VQs are locally lattice-similar, but globally optimized to the input pdf. The design algorithm was complemented with a new lattice-based fast search algorithm. Experiments on Gaussian data with the proposed fast search algorithm illustrated that the performance is excellent, with only moderate extra storage requirements.

APPENDIX A

In this appendix, theorem I (3.13) and theorem II (3.14) in section 3.2 are proved. In section A.1, some definitions and preliminaries are presented. Section A.2 discusses the overload distortion (theorem I), and section A.3 treats the granular distortion (theorem II). In section A.4, the total distortion, which is the sum of overload and granular distortion, is treated, and methods to find the global minimum is discussed.

A.1 Preliminaries

For the proofs in the appendix, we use the definition of a *d*-sphere (3.9), the truncation radius a_{T} (3.10), and the granular region $\mathcal{G}$ (3.11), all defined in section 3.2. We also use the VQ definitions in chapter 2, and the lattice definitions in section 3.1, together with some new definitions in this section. As discussed in section 3.2, we assume zero mean, iid Gaussian variables, with unit variance samples.

A *granular Voronoi region* $\Omega_{\mathcal{G}}(\mathbf{c})$ is defined as the lattice Voronoi region Ω, translated to the codevector **c**,

$$\Omega_{\mathcal{G}}(\mathbf{c}) = \Omega + \mathbf{c} = \Omega(\mathbf{c}) \cap \mathcal{G}, \tag{A.1}$$

where $\Omega(\mathbf{c})$ is the Voronoi region around codevector **c** (see 2.7), Ω is the lattice Voronoi region (see 3.4), and $\mathcal{G}$ is the granular region (see 3.11).

For a given input vector **x,** we define $\mathbf{p}(a)$ as the closest point to **x** in a sphere with radius *a*,

$$\mathbf{p}(a) = \underset{\mathbf{y}:\|\mathbf{y}\|<a}{\operatorname{argmin}} \|\mathbf{x}-\mathbf{y}\| = \begin{cases} \mathbf{x} & \|\mathbf{x}\| \le a \\ a \cdot \dfrac{\mathbf{x}}{\|\mathbf{x}\|} & \|\mathbf{x}\| > a \end{cases} \tag{A.2}$$

With this definition, the distance between $\mathbf{x}$ and $\mathbf{p}(a)$ is given by

$$\|\mathbf{x}-\mathbf{p}(a)\| = \max\bigl(0, \|\mathbf{x}\| - a\bigr). \tag{A.3}$$

We define the *granular radius* $a_{\mathcal{G}}$ as the effective radius of the granular region $\mathcal{G}$,

$$a_{\mathcal{G}} = \left(\frac{\mathrm{vol}(\mathcal{G})}{\mathrm{vol}(S_d(1))} \right)^{1/d}, \tag{A.4}$$

where $S_d(\phi)$ is a sphere with radius ϕ, see (3.9). The volume of the sphere $S_d(a_{\mathcal{G}})$, called the *granular sphere*, is with this definition equal to the volume of the granular region, i.e. $\mathrm{vol}(S_d(a_{\mathcal{G}})) = \mathrm{vol}(\mathcal{G})$. The granular radius $a_{\mathcal{G}}$ and the truncation radius a_{T} are closely related, and we show in (A.24) that they are equal for infinite rates.

We define a *border region* $\mathcal{B}$ in the form of a spherical shell (see figure A.1),

$$\mathcal{B} = \left\{ \mathbf{x} \in \mathbb{R}^d : a_{\min} < \|\mathbf{x}\| \leq a_{\max} \right\} \tag{A.5}$$

which overlaps both the granular and the overload region. The border shell is defined as the thinnest shell having only granular region on the inside and only overload region on the outside, that is, $a_{\min}$ is the radius of the inscribed sphere, and $a_{\max}$ is the radius of the circumscribed sphere of the granular region,

$$a_{\min} = \inf_{\mathbf{x} \in \overline{\mathcal{G}}} \|\mathbf{x}\| \tag{A.6}$$

$$a_{\max} = \sup_{\mathbf{x} \in \mathcal{G}} \|\mathbf{x}\|. \tag{A.7}$$

The border region is a mix of granular and overload regions. Figure A.1 illustrates the border region for a two-dimensional lattice VQ.

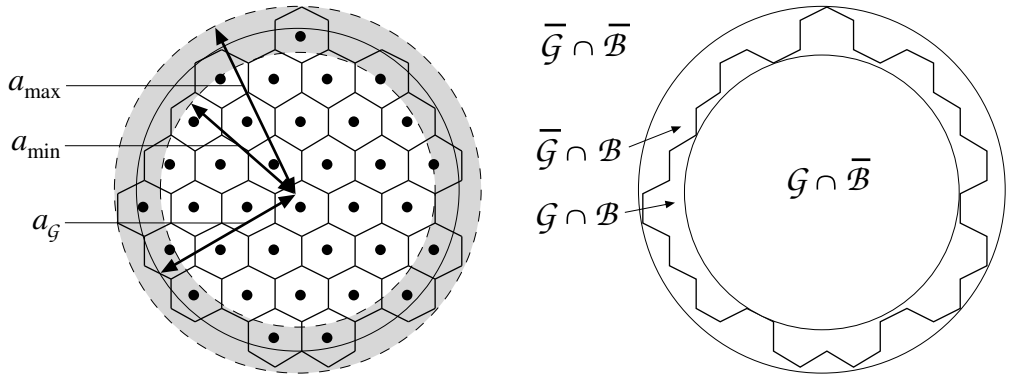


Figure A.1. Left: Illustration of the border region (the gray area). Right: Combinations of the granular and the border region.

With the definitions of overload and granular regions in (3.11), and the border region in (A.5), we have

$$\overline{\mathcal{G}} \cap \overline{\mathcal{B}} = \left\{ \mathbf{x} \in \mathbb{R}^d : \|\mathbf{x}\| > a_{\max} \right\} \subseteq \overline{\mathcal{G}} \subseteq \left\{ \mathbf{x} \in \mathbb{R}^d : \|\mathbf{x}\| > a_{\min} \right\} = \overline{\mathcal{G}} \cup \mathcal{B} \tag{A.8}$$

$$\mathcal{G} \cap \overline{\mathcal{B}} = \left\{ \mathbf{x} \in \mathbb{R}^d : \|\mathbf{x}\| \leq a_{\min} \right\} \subseteq \mathcal{G} \subseteq \left\{ \mathbf{x} \in \mathbb{R}^d : \|\mathbf{x}\| \leq a_{\max} \right\} = \mathcal{G} \cup \mathcal{B}. \tag{A.9}$$

From (A.9), we conclude that the radius of the granular sphere, $a_{\mathcal{G}}$, is bounded between $a_{\min}$ and $a_{\max}$, since

$$\mathrm{vol}\left(S_d(a_{\min})\right) \le \mathrm{vol}(\mathcal{G}) = \mathrm{vol}\left(S_d\left(a_{\mathcal{G}}\right)\right) \le \mathrm{vol}\left(S_d(a_{\max})\right) \Rightarrow a_{\min} \le a_{\mathcal{G}} \le a_{\max}. \quad \text{(A.10)}$$

We use the covering radius $r_{\max}$, the packing radius $r_{\min}$, and the effective radius r_Ω of a granular Voronoi region $\Omega_{\mathcal{G}}(\mathbf{c})$, defined as

$$r_{\max} = \sup_{\mathbf{x}\in\Omega_{\mathcal{G}}(\mathbf{c})} \|\mathbf{x}-\mathbf{c}\| = \sup_{\mathbf{x}\in\Omega} \|\mathbf{x}\| \quad \text{(A.11)}$$

$$r_{\min} = \inf_{\mathbf{x}\notin\Omega_{\mathcal{G}}(\mathbf{c})} \|\mathbf{x}-\mathbf{c}\| = \inf_{\mathbf{x}\notin\Omega} \|\mathbf{x}\| \quad \text{(A.12)}$$

$$r_\Omega = \left(\frac{\mathrm{vol}(\Omega)}{\mathrm{vol}\left(S_d(1)\right)}\right)^{1/d} = \left(\frac{\mathrm{vol}(\mathcal{G})}{N\cdot\mathrm{vol}\left(S_d(1)\right)}\right)^{1/d} = \left(\frac{\mathrm{vol}\left(S_d\left(a_{\mathcal{G}}\right)\right)}{N\cdot\mathrm{vol}\left(S_d(1)\right)}\right)^{1/d} = a_{\mathcal{G}}\cdot 2^{-R}. \quad \text{(A.13)}$$

The three radii, r_Ω, $r_{\min}$ and $r_{\max}$, are illustrated in figure A.2.

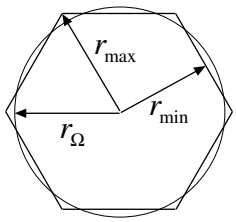


Figure A.2. A Voronoi region.

The granular Voronoi regions are all bounded and congruent, and thus the ratios $r_{\max}/r_\Omega$ and $r_{\min}/r_\Omega$ are bounded, nonzero and independent of the scaling of the region, so that

$$r_{\max} = \frac{r_{\max}}{r_\Omega}\cdot r_\Omega = \frac{r_{\max}}{r_\Omega}\cdot a_{\mathcal{G}}\cdot 2^{-R} \quad \text{(A.14)}$$

$$r_{\min} = \frac{r_{\min}}{r_\Omega}\cdot r_\Omega = \frac{r_{\min}}{r_\Omega}\cdot a_{\mathcal{G}}\cdot 2^{-R}. \quad \text{(A.15)}$$

$a_{\min}$ and $a_{\max}$ can be bounded as (using the definition of Λ in (3.1))

$$\begin{aligned} a_{\min} &= \inf_{\mathbf{x}\in\overline{\mathcal{G}}} \|\mathbf{x}\| = \inf_{\mathbf{x}\in \bigcup_{\mathbf{c}_i\in\Lambda\setminus\mathcal{C}}(\Omega+\mathbf{c}_i)} \|\mathbf{x}\| = \inf_{\mathbf{c}_i\in\Lambda\setminus\mathcal{C}} \inf_{\mathbf{x}\in(\Omega+\mathbf{c}_i)} \|\mathbf{x}\| = \inf_{\mathbf{c}_i\in\Lambda\setminus\mathcal{C}} \inf_{\mathbf{x}\in\Omega} \|\mathbf{x}-\mathbf{c}_i\| \ge \\ &\ge \inf_{\mathbf{c}_i\in\Lambda\setminus\mathcal{C}} \inf_{\mathbf{x}\in\Omega} \left(\|\mathbf{c}_i\| - \|\mathbf{x}\|\right) = \inf_{\mathbf{c}_i\in\Lambda\setminus\mathcal{C}} \|\mathbf{c}_i\| - \sup_{\mathbf{x}\in\Omega} \|\mathbf{x}\| \ge a_{\mathrm{T}} - r_{\max}, \end{aligned} \quad \text{(A.16)}$$

$$\begin{aligned} a_{\max} &= \sup_{\mathbf{x}\in\mathcal{G}} \|\mathbf{x}\| = \sup_{\mathbf{x}\in \bigcup_{\mathbf{c}_i\in\mathcal{C}}(\Omega+\mathbf{c}_i)} \|\mathbf{x}\| = \sup_{\mathbf{c}_i\in\mathcal{C}} \sup_{\mathbf{x}\in(\Omega+\mathbf{c}_i)} \|\mathbf{x}\| = \sup_{\mathbf{c}_i\in\mathcal{C}} \sup_{\mathbf{x}\in\Omega} \|\mathbf{x}-\mathbf{c}_i\| \le \\ &\le \sup_{\mathbf{c}_i\in\mathcal{C}} \sup_{\mathbf{x}\in\Omega} \left(\|\mathbf{x}\| + \|\mathbf{c}_i\|\right) = \sup_{\mathbf{c}_i\in\mathcal{C}} \|\mathbf{c}_i\| + \sup_{\mathbf{x}\in\Omega} \|\mathbf{x}\| \le a_{\mathrm{T}} + r_{\max}. \end{aligned} \quad \text{(A.17)}$$

where the last inequality of (A.16) and (A.17) follows from the truncation of the lattice by a hypersphere with radius a_{T}, as in (3.10). Now, using (A.10), (A.16) and (A.17), we can bound the truncation radius a_{T} as

$$a_{\mathcal{G}} - r_{\max} \le a_{\max} - r_{\max} \le a_{\mathrm{T}} \le a_{\min} + r_{\max} \le a_{\mathcal{G}} + r_{\max}. \tag{A.18}$$

The left- and right-most terms of (A.18) can both be written[11] $a_{\mathcal{G}} + r_{\max} \cdot O(1)$, and using (A.14), we get

$$a_{\mathrm{T}} = a_{\mathcal{G}} + r_{\max} \cdot O(1) = a_{\mathcal{G}} \cdot \left(1 + 2^{-R} \cdot O(1)\right). \tag{A.19}$$

Using (A.19) to eliminate $a_{\mathcal{G}}$ from (A.13)-(A.15), we get the useful equalities

$$r_{\max} = a_{\mathrm{T}} \cdot 2^{-R} \cdot O(1) \tag{A.20}$$
$$r_{\Omega} = a_{\mathrm{T}} \cdot 2^{-R} \cdot O(1) \tag{A.21}$$
$$r_{\min} = a_{\mathrm{T}} \cdot 2^{-R} \cdot O(1), \tag{A.22}$$

and by inserting (A.20) into (A.18), we get

$$a_{\min} = a_{\mathrm{T}} \cdot \left(1 + 2^{-R} \cdot O(1)\right) \tag{A.23}$$
$$a_{\mathcal{G}} = a_{\mathrm{T}} \cdot \left(1 + 2^{-R} \cdot O(1)\right) \tag{A.24}$$
$$a_{\max} = a_{\mathrm{T}} \cdot \left(1 + 2^{-R} \cdot O(1)\right), \tag{A.25}$$

which illustrates that a_{T}, $a_{\mathcal{G}}$, $a_{\min}$, and $a_{\max}$ are all equal for infinite rates.

A.2 Theorem I: Overload distortion

In theorem I in section 3.2, we stated that the overload distortion is given by

$$D_{\overline{\mathcal{G}}} = f_{\overline{\mathcal{G}}}(d) \cdot a_{\mathrm{T}}^{d-4} \cdot e^{-a_{\mathrm{T}}^2/2} \cdot \left(1 + \varepsilon_{\overline{\mathcal{G}}}\right), \tag{A.26}$$

where $f_{\overline{\mathcal{G}}}(d) = \left(2^{d/2-2} \cdot \Gamma(d/2)\right)^{-1}$, and $\varepsilon_{\overline{\mathcal{G}}}$ tends to zero for asymptotically high rates R. In this section, we present a proof of this theorem. In the proof, we bound the overload distortion by use of two spheres, one outside and one inside the border region. Then we complete the proof by showing that the width of the border region tends to zero when the rate approaches infinity.

We write the overload distortion

$$D_{\overline{\mathcal{G}}} = \int_{\overline{\mathcal{G}}} \left\| \mathbf{x} - \mathbf{c}^* \right\|^2 f_{\mathbf{x}}(\mathbf{x}) d\mathbf{x}, \tag{A.27}$$

where $\mathbf{c}^*$ is the codevector in the codebook $\mathcal{C}$ that is closest to the input vector $\mathbf{x}$, and $f_{\mathbf{x}}(\mathbf{x})$ is the input pdf. The integrand is nonnegative, so we can lower- and upper-bound the distortion by integrating over a smaller and larger region, respectively. Using (A.8), we get

$$\int_{\|\mathbf{x}\| > a_{\max}} \left\| \mathbf{x} - \mathbf{c}^* \right\|^2 f_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} \le D_{\overline{\mathcal{G}}} \le \int_{\|\mathbf{x}\| > a_{\min}} \left\| \mathbf{x} - \mathbf{c}^* \right\|^2 f_{\mathbf{x}}(\mathbf{x}) d\mathbf{x}. \tag{A.28}$$

[11] With $g \cdot O(1)$ (big-oh), we will mean $g \cdot C$, where C is bounded in a neighborhood of $g = 0$. Rules for computation using big-oh can be found in most mathematical handbooks, e.g. [49].

We now study the upper and lower bound in (A.28) separately. First, noting that all codevectors lie inside a sphere with radius $a_{\max}$, we can lower-bound the integrand

$$\left\| \mathbf{x} - \mathbf{c}^* \right\| = \min_{\mathbf{c}\in\mathcal{C}} \| \mathbf{x} - \mathbf{c} \| \geq \min_{\|\mathbf{y}\|<a_{\max}} \| \mathbf{x} - \mathbf{y} \| = \left\| \mathbf{x} - \mathbf{p}(a_{\max}) \right\|. \tag{A.29}$$

Secondly, the integrand can be upper-bounded by use of the triangle inequality,

$$\left\| \mathbf{x} - \mathbf{c}^* \right\| \leq \left\| \mathbf{x} - \mathbf{p}(a_{\min}) \right\| + \left\| \mathbf{p}(a_{\min}) - \mathbf{c}^* \right\|. \tag{A.30}$$

With the definition of $a_{\min}$ in (A.6), $\mathbf{p}(a_{\min})$ belongs to a granular Voronoi region. Therefore, we can bound the distance between $\mathbf{p}(a_{\min})$ and $\mathbf{c}^*$ by the covering radius of the Voronoi region, $r_{\max}$,

$$\left\| \mathbf{p}(a_{\min}) - \mathbf{c}^* \right\| \leq r_{\max} \tag{A.31}$$

(see (A.11) and figure A.2). Thus, we have

$$\left\| \mathbf{x} - \mathbf{c}^* \right\| \leq \| \mathbf{x} \| - a_{\min} + r_{\max} = \left\| \mathbf{x} - \mathbf{p}(a_{\min} - r_{\max}) \right\| \quad \text{if } \| \mathbf{x} \| > a_{\min}, \tag{A.32}$$

where we have also used (A.3). The distortion upper bound is

$$D_{\bar{\mathcal{G}}} \leq \int_{\|\mathbf{x}\|>a_{\min}} \left\| \mathbf{x} - \mathbf{p}(a_{\min} - r_{\max}) \right\|^2 f_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} \leq \int_{\|\mathbf{x}\|>a_{\min}-r_{\max}} \left\| \mathbf{x} - \mathbf{p}(a_{\min} - r_{\max}) \right\|^2 f_{\mathbf{x}}(\mathbf{x}) d\mathbf{x}. \tag{A.33}$$

Combining (A.28), (A.29) and (A.33), we get

$$\int_{\|\mathbf{x}\|>a_{\max}} \left\| \mathbf{x} - \mathbf{p}(a_{\max}) \right\|^2 f_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} \leq D_{\bar{\mathcal{G}}} \leq \int_{\|\mathbf{x}\|>a_{\min}-r_{\max}} \left\| \mathbf{x} - \mathbf{p}(a_{\min} - r_{\max}) \right\|^2 f_{\mathbf{x}}(\mathbf{x}) d\mathbf{x}, \tag{A.34}$$

which bounds the overload distortion by use of two spheres with radii $a_{\max}$ and $a_{\min} - r_{\max}$. From (A.20), (A.23) and (A.25), we see that both radii can be written on the same form, $a_{\mathrm{T}} \cdot \left(1 + 2^{-R} \cdot O(1)\right)$. We define

$$\hat{a} = a_{\mathrm{T}} \cdot \left(1 + 2^{-R} \cdot O(1)\right), \tag{A.35}$$

and rewrite the overload distortion as

$$\begin{aligned} D_{\bar{\mathcal{G}}} &= \int_{\|\mathbf{x}\|>\hat{a}} \left\| \mathbf{x} - \mathbf{p}(\hat{a}) \right\|^2 f_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} && \text{(A.36)} \\ &= \int_{\|\mathbf{x}\|>\hat{a}} \left(\|\mathbf{x}\| - \hat{a} \right)^2 f_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} && \text{(A.37)} \\ &= \int_{\|\mathbf{x}\|>\hat{a}} \left(\|\mathbf{x}\|^2 + \hat{a}^2 - 2\hat{a}\|\mathbf{x}\| \right) f_{\mathbf{x}}(\mathbf{x}) d\mathbf{x}. && \text{(A.38)} \end{aligned}$$

Now the *d*-dimensional integral has become one-dimensional; the integrand is a function of $\|\mathbf{x}\|$ only[12]. The stochastic variable $\xi = \|\mathbf{x}\|^2$ has a χ^2-distribution with *d* degrees of freedom, $f_\xi(\xi) = \chi^2(d, \xi)$, and we get

[12]Since the Gaussian pdf $f_{\mathbf{x}}(\mathbf{x})$ is spherically symmetrical, it is a function of $\|\mathbf{x}\|$ only.

$$D_{\bar{\mathcal{G}}} = \int_{\hat{a}^2}^{\infty} \left(\xi + \hat{a}^2 - 2\hat{a}\sqrt{\xi}\right)\chi^2(d,\xi)d\xi \tag{A.39}$$

$$= \int_{\hat{a}^2}^{\infty} \left(\xi + \hat{a}^2 - 2\hat{a}\sqrt{\xi}\right)\frac{\xi^{d/2-1}\cdot e^{-\xi/2}}{2^{d/2}\cdot\Gamma(d/2)}d\xi. \tag{A.40}$$

In the sequel, we need the incomplete Gamma function,

$$\Gamma(b,z) = \int_{z}^{\infty} t^{b-1}e^{-t}dt. \tag{A.41}$$

Using $\Gamma(b,z)$, we write the overload distortion as

$$D_{\bar{\mathcal{G}}} = \frac{1}{\Gamma(d/2)}\cdot\left[2\cdot\Gamma\left(\frac{d+2}{2},\frac{\hat{a}^2}{2}\right) - 2\sqrt{2}\cdot\hat{a}\cdot\Gamma\left(\frac{d+1}{2},\frac{\hat{a}^2}{2}\right) + \hat{a}^2\cdot\Gamma\left(\frac{d}{2},\frac{\hat{a}^2}{2}\right)\right]. \tag{A.42}$$

We approximate the incomplete Gamma function as an asymptotic series [50]:

$$\Gamma(b,z) = z^{b-1}e^{-z}\left[1 + (b-1)z^{-1} + (b-1)(b-2)z^{-2} + z^{-3}O(1)\right]. \tag{A.43}$$

With this approximation, the overload distortion can, after some work, be written

$$D_{\bar{\mathcal{G}}} = \frac{\hat{a}^{d-4}e^{-\hat{a}^2/2}}{2^{d/2-2}\cdot\Gamma(d/2)}\cdot\left(1 + \hat{a}^{-2}\cdot O(1)\right) \tag{A.44}$$

Insertion of (A.35) yields, again omitting the details,

$$D_{\bar{\mathcal{G}}} = \frac{a_{\mathrm{T}}^{d-4}\cdot e^{-a_{\mathrm{T}}^2/2}}{2^{d/2-2}\cdot\Gamma(d/2)}\cdot\left[1 + a_{\mathrm{T}}^2\cdot 2^{-R}\cdot O(1) + a_{\mathrm{T}}^{-2}\cdot O(1)\right], \tag{A.45}$$

which is equal to (A.26), and the proof is completed.

In section A.4, we will verify that the error term is equal to zero for asymptotically high rates if the truncation radius is selected for minimum distortion.

A.3 Theorem II: Granular distortion

In theorem II, the granular distortion is given by

$$D_{\mathcal{G}} = f_{\mathcal{G}}(d)\cdot 2^{-2R}\cdot a_{\mathrm{T}}^2\cdot\left(1+\varepsilon_{\mathcal{G}}\right) \tag{A.46}$$

where $f_{\mathcal{G}}(d) = G\cdot d\cdot\pi\cdot\Gamma(d/2+1)^{-2/d}$, and $\varepsilon_{\mathcal{G}}$ tends to zero for asymptotically high rates *R*. The proof of the theorem, which is given in this section, is based on writing the pdf inside each Voronoi region as a uniform pdf plus an error term. The granular distortion for a uniform pdf is easily computed, and the proof is completed by showing that the error term is zero for infinite rates.

We write the granular distortion for the *N*-point lattice VQ as a sum of the Voronoi region distortions

$$D_{\mathcal{G}} = \int_{\mathcal{G}} \left\| \mathbf{x} - \mathbf{c}^* \right\| f_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} \tag{A.47}$$

$$= \sum_{k=1}^{N} \int_{\Omega_{\mathcal{G}}(\mathbf{c}_k)} \left\| \mathbf{x} - \mathbf{c}_k \right\| f_{\mathbf{x}}(\mathbf{x}) d\mathbf{x}. \tag{A.48}$$

For bounded and differentiable densities, we can expand the pdf in a Taylor series as

$$f_{\mathbf{x}}(\mathbf{x}) = f_{\mathbf{x}}(\mathbf{c}_k) + \left\| \mathbf{x} - \mathbf{c}_k \right\| \cdot O(1), \tag{A.49}$$

and (A.48) can be rewritten as

$$D_{\mathcal{G}} = \sum_{k=1}^{N} \left(\int_{\Omega_{\mathcal{G}}(\mathbf{c}_k)} \left\| \mathbf{x} - \mathbf{c}_k \right\|^2 \left(f_{\mathbf{x}}(\mathbf{c}_k) + \left\| \mathbf{x} - \mathbf{c}_k \right\| \cdot O(1) \right) d\mathbf{x} \right) \tag{A.50}$$

$$= \sum_{k=1}^{N} \left(f_{\mathbf{x}}(\mathbf{c}_k) \cdot \int_{\Omega_{\mathcal{G}}(\mathbf{c}_k)} \left\| \mathbf{x} - \mathbf{c}_k \right\|^2 d\mathbf{x} \right) + \sum_{k=1}^{N} \left(\int_{\Omega_{\mathcal{G}}(\mathbf{c}_k)} \left\| \mathbf{x} - \mathbf{c}_k \right\|^3 d\mathbf{x} \cdot O(1) \right). \tag{A.51}$$

Now, since the granular Voronoi regions $\Omega_{\mathcal{G}}(\mathbf{c}_k)$ are congruent, the integrals in (A.51) are independent of *k*, and we get

$$D_{\mathcal{G}} = \int_{\Omega} \left\| \mathbf{x} \right\|^2 d\mathbf{x} \cdot \sum_{k=1}^{N} f_{\mathbf{x}}(\mathbf{c}_k) + \sum_{k=1}^{N} \int_{\Omega} \left\| \mathbf{x} \right\|^3 d\mathbf{x} \cdot O(1). \tag{A.52}$$

The first integral in (A.52) is recognized to be a scaled version of the lattice quantization constant *G* (3.6). The second integral can be simplified by using (A.11), and writing $\left\| \mathbf{x} \right\| = r_{\max} \cdot O(1)$. We get

$$D_{\mathcal{G}} = d \cdot \mathrm{vol}(\Omega)^{1+2/d} \cdot G \cdot \sum_{k=1}^{N} f_{\mathbf{x}}(\mathbf{c}_k) + r_{\max}^3 \cdot \mathrm{vol}(\mathcal{G}) \cdot O(1) \tag{A.53}$$

$$= d \cdot \mathrm{vol}(\Omega)^{1+2/d} \cdot G \cdot \sum_{k=1}^{N} f_{\mathbf{x}}(\mathbf{c}_k) + a_{\mathrm{T}}^{d+3} \cdot 2^{-3R} \cdot O(1), \tag{A.54}$$

where (A.20) is used for the last equality.

The sum in (A.54) is considered next. For this reason, we study the granular probability $\Pr(\mathbf{x} \in \mathcal{G})$. Using the same approach as in (A.47)-(A.54), we can write the granular probability

$$\Pr(\mathbf{x} \in \mathcal{G}) = \int_{\mathcal{G}} f_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} \tag{A.55}$$

$$= \mathrm{vol}(\Omega) \cdot \sum_{k=1}^{N} f_{\mathbf{x}}(\mathbf{c}_k) + a_{\mathrm{T}}^{d+1} \cdot 2^{-R} \cdot O(1). \tag{A.56}$$

We can also write the granular probability using the overload probability, as

$$\Pr(\mathbf{x} \in \mathcal{G}) = 1 - \Pr(\mathbf{x} \in \overline{\mathcal{G}}). \tag{A.57}$$

Using (A.8), we can bound the overload probability as

$$\Pr\left(\mathbf{x} \in \overline{\mathcal{G}}\right) \le \Pr\left(\|\mathbf{x}\| > a_{\min}\right). \tag{A.58}$$

(A.58) can be written using the χ^2-distribution as in (A.39). We get

$$\Pr\left(\mathbf{x} \in \overline{\mathcal{G}}\right) \le \frac{\Gamma\left(d/2,\, a_{\min}^2/2\right)}{\Gamma\left(d/2\right)}, \tag{A.59}$$

which can be simplified using the first term in (A.43),

$$\Pr\left(\mathbf{x} \in \overline{\mathcal{G}}\right) = a_{\min}^{d-2} \cdot e^{-a_{\min}^2/2} \cdot O(1) = a_{\mathrm{T}}^{d-2} \cdot e^{-a_{\mathrm{T}}^2/2} \cdot O(1) \tag{A.60}$$

(see (A.23)). Combining (A.56), (A.57) and (A.60), we get

$$\mathrm{vol}(\Omega) \cdot \sum_{k=1}^{N} f_{\mathbf{x}}\left(\mathbf{c}_k\right) = 1 + a_{\mathrm{T}}^{d+1} \cdot 2^{-R} \cdot O(1) + a_{\mathrm{T}}^{d-2} \cdot e^{-a_{\mathrm{T}}^2/2} \cdot O(1). \tag{A.61}$$

Using the number of codevectors in the quantizer, $N = 2^{R \cdot d}$, the volume of the Voronoi region, $\mathrm{vol}(\Omega)$, can be written

$$\mathrm{vol}(\Omega) = \frac{\mathrm{vol}\left(\mathcal{G}\right)}{N} = \frac{\mathrm{vol}\left(S_d\left(a_{\mathcal{G}}\right)\right)}{N} = \frac{\pi^{d/2} \cdot a_{\mathcal{G}}^d \cdot 2^{-Rd}}{\Gamma\left(d/2+1\right)}, \tag{A.62}$$

where we have used the fact that the volume of the granular region, $\mathrm{vol}\left(\mathcal{G}\right)$, is equal to the volume of a *d*-sphere [50] with radius $a_{\mathcal{G}}$, see (A.10). Inserting (A.24), the volume of the lattice Voronoi region is expressed as a function of the truncation radius a_{T},

$$\mathrm{vol}(\Omega) = \frac{\pi^{d/2} \cdot a_{\mathrm{T}}^d \cdot 2^{-Rd}}{\Gamma\left(d/2+1\right)} \cdot \left(1 + 2^{-R} \cdot O(1)\right). \tag{A.63}$$

Inserting (A.61) and (A.63) into (A.54), we get

$$D_{\mathcal{G}} = \frac{G \cdot d \cdot \pi}{\Gamma^{2/d}(d/2+1)} \cdot a_{\mathrm{T}}^2 \cdot 2^{-2R} \cdot \left[1 + a_{\mathrm{T}}^{d+1} \cdot 2^{-R} \cdot O(1) + a_{\mathrm{T}}^{d-2} \cdot e^{-a_{\mathrm{T}}^2/2} \cdot O(1)\right], \tag{A.64}$$

which equals (A.46). If the error terms in (A.64) are excluded, the equation describes the distortion for quantization of a spherical uniform pdf (see [14], (1.10)).

In section A.4, we show that for an optimal choice of a_{T}, the error terms tend to zero for a rate approaching infinity.

A.4 Total distortion

The key issue in the high rate theory is to find the optimal value of the truncation radius a_{T}. We study three possible choices of a_{T}:

I a_{T} does not grow towards infinity with the rate.

II a_{T} grows towards infinity with the rate, but slower than exponentially in *R*.

III a_{T} grows towards infinity exponentially in R, or even faster, i.e. $a_{\mathrm{T}} \geq 2^{\lambda R}$ for some λ.

We show in the following that I and III lead to higher distortion than II. For this reason, we use an arbitrary formula for a_{T} fulfilling II, and compute the resulting distortion. Then we lower-bound the distortion in I and III by simple calculations. The proof is completed by showing that the distortion for case II is lower than the lower bounds of distortion for case I and III.

First we study the distortion for case II above. For this case, the error terms in (A.26), (A.45) and (A.46), (A.64) are zero for asymptotically high rates. The total distortion is the sum of (A.26) and (A.46),

$$D = \left(f_{\mathcal{G}} \cdot a_{\mathrm{T}}^2 \cdot 2^{-2R} + f_{\overline{\mathcal{G}}} \cdot a_{\mathrm{T}}^{d-4} \cdot e^{-a_{\mathrm{T}}^2/2} \right)(1+\varepsilon). \tag{A.65}$$

We select the truncation radius arbitrarily as $a_{\mathrm{T}} = R$, which fulfills II. Insertion of a_{T} in (A.65) yields

$$D_{\mathrm{II}} = \left(f_{\mathcal{G}} \cdot R^2 \cdot 2^{-2R} + f_{\overline{\mathcal{G}}} \cdot R^{d-4} \cdot e^{-R^2/2} \right)(1+\varepsilon) \tag{A.66}$$

$$= f_{\mathcal{G}} \cdot R^2 \cdot 2^{-2R} \cdot \left(1 + R^{d-4} \cdot e^{-R^2/2} \cdot 2^{2R} \cdot O(1) \right)(1+\varepsilon) \tag{A.67}$$

$$= f_{\mathcal{G}} \cdot R^2 \cdot 2^{-2R} \cdot \left(1 + R^{d-4} \cdot e^{-R^2/2 + R \cdot 2 \cdot \ln 2} \cdot O(1) \right)(1+\varepsilon). \tag{A.68}$$

The error terms are zero for infinite rates. We write

$$D_{\mathrm{II}} = R^2 \cdot 2^{-2R} \cdot O(1), \tag{A.69}$$

and observe that the distortion tends to zero when the rate approaches infinity. Since we have used an arbitrary truncation radius fulfilling II, the optimal truncation radius gives a distortion lower than or equal to (A.69).

Now we study case I. In (A.34) a lower bound for the overload distortion is given. Using (A.17) we get

$$D_{\mathrm{I}} \geq D_{\overline{\mathcal{G}}} \geq \int_{\|\mathbf{x}\| > a_{\max}} \left\| \mathbf{x} - \mathbf{p}(a_{\max}) \right\|^2 f_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} \geq \int_{\|\mathbf{x}\| > a_{\mathrm{T}} + r_{\max}} \left\| \mathbf{x} - \mathbf{p}(a_{\max}) \right\|^2 f_{\mathbf{x}}(\mathbf{x}) d\mathbf{x}. \tag{A.70}$$

We observe that, for finite a_{T} and $r_{\max}$, the right-hand integral in (A.70) does not tend to zero as the rate approaches infinity. Since a_{T} is finite in case I, and $r_{\max}$ is finite for finite a_{T} (A.20), we conclude that D_{I} does not tend to zero as the rate approaches infinity. But $D_{\mathrm{II}} \to 0$ for $R \to \infty$, and we have shown that the optimal high-rate distortion in case I is higher than the distortion in case II, i.e. $D_{\mathrm{I}} > D_{\mathrm{II}}$.

To lower-bound the distortion in case III, we first define a shape $\mathcal{S}$ in the form of a d-sphere from which we cut out spherical holes around all codevectors $\mathbf{c}$,

$$\mathcal{S} = S_d(\beta) \setminus \bigcup_{\mathbf{c}\in\mathcal{C}} \left(S_d(\alpha \cdot r_{\min}) + \mathbf{c}\right), \tag{A.71}$$

where $0 < \alpha < 1$, and the radius β is an arbitrary constant, independent of R. $\mathcal{S}$ is illustrated in figure A.3.

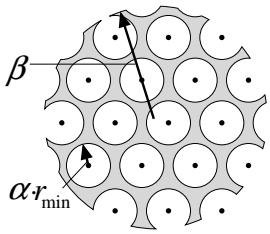


Figure A.3. The hollow shape $\mathcal{S}$.

Since α is less than 1, the definition of $r_{\min}$ (A.12) ensures that the holes, with radius $\alpha \cdot r_{\min}$, are nonoverlapping. Further, since the truncation radius a_{T} (and $a_{\min}$, see (A.23)) grows towards infinity with the rate, there exists a constant R_0 such that for all rates $R > R_0$, $a_{\min} > \beta$, which makes $\mathcal{S}$ a subset of the granular region $\mathcal{G}$. We have

$$D_{\mathrm{III}} \geq D_{\mathcal{G}} = \int_{\mathcal{G}} \left\| \mathbf{x} - \mathbf{c}^* \right\|^2 f_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} \geq \int_{S} \left\| \mathbf{x} - \mathbf{c}^* \right\|^2 f_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} \quad \text{for } R > R_0. \tag{A.72}$$

For vectors $\mathbf{x}$ in $\mathcal{S}$, the distance to the closest codeword $\mathbf{c}^*$ is lower-bounded by $\alpha \cdot r_{\min}$. The pdf $f_{\mathbf{x}}(\mathbf{x})$ is lower-bounded by the pdf at an arbitrary point at the surface of $\mathcal{S}$, i.e. $f_{\mathbf{x}}(\mathbf{x}) \leq f_{\mathbf{x}}(\mathbf{x}_\beta)$ where $\|\mathbf{x}_\beta\| = \beta$. Thus, for $R > R_0$, we have that (using (A.18))

$$D_{\mathrm{III}} \geq \int_{S} (\alpha \cdot r_{\min})^2 f_{\mathbf{x}}(\mathbf{x}_\beta) d\mathbf{x} \tag{A.73}$$

$$= (\alpha \cdot r_{\min})^2 \cdot f_{\mathbf{x}}(\mathbf{x}_\beta) \cdot \mathrm{vol}(\mathcal{S}) \tag{A.74}$$

$$= f_{\mathbf{x}}(\mathbf{x}_\beta) \cdot \mathrm{vol}(\mathcal{S}) \cdot (\alpha \cdot r_{\min}/r_\Omega)^2 \cdot a_{\mathcal{G}}^2 \cdot 2^{-2R} \tag{A.75}$$

$$\geq f_{\mathbf{x}}(\mathbf{x}_\beta) \cdot \mathrm{vol}(\mathcal{S}) \cdot (\alpha \cdot r_{\min}/r_\Omega)^2 \cdot (a_{\mathrm{T}} - r_{\max})^2 \cdot 2^{-2R} \tag{A.76}$$

$$\geq C \cdot a_{\mathrm{T}}^2 \cdot 2^{-2R}, \tag{A.77}$$

where C is a positive constant, since $r_{\max}/a_{\mathrm{T}}$ tends to zero (see (A.20)), and the volume of $\mathcal{S}$, the pdf $f_{\mathbf{x}}(\mathbf{x}_\beta)$ at the surface, and $r_{\min}/r_\Omega$ are all positive constants. Now, inserting a_{T} as in case III yields

$$D_{\mathrm{III}} \geq C \cdot 2^{2\lambda R} \cdot 2^{-2R} > D_{\mathrm{II}} \text{ for R} \to \infty, \tag{A.78}$$

and we have shown that radius selection as in case II leads to lower distortion than case III.

We will now study the total distortion, D, and show that, for a selection of a_{T} with the restrictions as in case II above, the distortion is convex and has a distinct global minimum. As discussed above, the error term in (A.65) is zero for infinite rate. We define $\hat{D}$ as D excluding the error term,

$$\hat{D} = f_{\mathcal{G}} \cdot a_{\mathrm{T}}^2 \cdot 2^{-2R} + f_{\bar{\mathcal{G}}} \cdot a_{\mathrm{T}}^{d-4} \cdot e^{-a_{\mathrm{T}}^2/2}. \qquad \text{(A.79)}$$

To show that $\hat{D}$ is convex with respect to a_{T}, we compute the second derivative of $\hat{D}$ with respect to a_{T}:

$$\frac{\partial^2 \hat{D}}{\partial a_{\mathrm{T}}^2} = 2 \cdot f_{\mathcal{G}} \cdot 2^{-2R} + f_{\bar{\mathcal{G}}} \cdot a_{\mathrm{T}}^{d-2} \cdot e^{-a_{\mathrm{T}}^2/2} \cdot \left[1 + (7-2d) \cdot a_{\mathrm{T}}^{-2} + \left(d^2 - 9d + 20\right) \cdot a_{\mathrm{T}}^{-4}\right]. \qquad \text{(A.80)}$$

We see that the expression inside brackets is dominated by the first term when a_{T} tends to infinity, and we write

$$\frac{\partial^2 \hat{D}}{\partial a_{\mathrm{T}}^2} = 2 \cdot f_{\mathcal{G}} \cdot 2^{-2R} + f_{\bar{\mathcal{G}}} \cdot a_{\mathrm{T}}^{d-2} \cdot e^{-a_{\mathrm{T}}^2/2} \cdot \left(1 + a_{\mathrm{T}}^{-2} \cdot O(1)\right). \qquad \text{(A.81)}$$

Clearly, this expression is positive for large enough values of a_{T}. Thus, $\hat{D}$ is a convex function of a_{T} in the region defined in case II, and the first derivative can only be zero at the global minimum of $\hat{D}$. D is the sum of $\hat{D}$ and error terms, but $\hat{D}$ dominates the distortion for all a_{T} satisfying case II, so the global minimum of $\hat{D}$ is the global minimum of D as well.

BIBLIOGRAPHY

[1] B.-H. Juang and A. H. Gray, "Multiple stage vector quantization for speech coding," *in Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, Paris, France, vol. 1, pp. 597-600, 1982.

[2] R. M. Gray and H. Abut, "Full search and tree searched vector quantization of speech waveforms," *in Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, Paris, France, pp. 593-596, 1982.

[3] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communications*, vol. 28, no. 1, pp. 84-95, January 1980.

[4] J. Makhoul, S. Roucos, and H. Gish, "Vector quantization in speech coding," *Proceedings of the IEEE*, vol. 73, no. 11, pp. 1551-1588, November 1985.

[5] A. Gersho and V. Cuperman, "Vector quantization: A pattern-matching technique for speech coding," *IEEE Communications Magazine*, vol. 21, no. 9, pp. 15-21, December 1983.

[6] I. A. Gerson and M. A. Jasiuk, "Vector sum excited linear prediction (VSELP) speech coding at 8 kbps," *in Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, Albuquerque, USA, pp. 461-464, 1990.

[7] M. J. Sabin and R. M. Gray, "Product code vector quantizers for speech and waveform coding," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 32, pp. 474-488, June 1984.

[8] J. D. Gibson and K. Sayood, "Lattice quantization," *Advances in Electronics and Electron Physics*, vol. 72, pp. 259-3321988.

[9] J. H. Conway and N. J. A. Sloane, *Sphere packings, lattices and groups*, Second ed., Springer-Verlag, 1992.

[10] P. J. Green and R. Sibson, "Computing Dirichlet tessellations in the plane," *The Computer Journal*, vol. 21, no. 2, pp. 168-173, May 1978.

[11] E. Agrell and P. Hedelin, "How to evaluate search methods for vector quantization," *in Proc. NORSIG*, Ålesund, Norway, pp. 258-263, 1994.

[12] V. Ramasubramanian and K. K. Paliwal, "An efficient approximation-elimination algorithm for fast nearest-neighbor search based on a spherical distance coordinate formulation," *Pattern Recognition Letters*, vol. 13, no. 7, pp. 471-480, July 1992.

[13] V. Ramasubramanian and K. K. Paliwal, "Fast K-dimensional tree algorithms for nearest neighbor search with application to vector quantization encoding," *IEEE Transactions on Signal Processing*, vol. 40, no. 3, pp. 518-531, March 1992.

[14] E. Agrell and T. Eriksson, "Lattice-based quantization, part I," Technical report no. 17, Chalmers University of Technology, October, 1996.

[15] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, no. 3, pp. 379-423, 623-656, July-October 1948.

[16] A. Gersho and R. M. Gray, *Vector quantization and signal compression*, Kluwer Academic Publishers, 1992.

[17] G. Voronoï, "Nouvelles applications des paramètres continus à la théorie des formes quadratiques," *Journal für die reine und angewandte Mathematik*, vol. 133, 134 and 1361908-1909.

[18] R. M. Gray, *Source coding theory*, Kluwer Academic Publishers, 1990.

[19] P. Knagenhjelm, *Competitive learning in robust communication*, PhD dissertation, Chalmers University of Technology, Gothenburg, Sweden, 1993.

[20] M. Antonini, M. Barlaud, and T. Gaidon, "Adaptive entropy constrained lattice vector quantization for multiresolution image coding," *Proceedings of SPIE Visual Communications and Image processing*, vol. 1818, no. 2, pp. 441-457, November 1992.

[21] A. Woolf and G. Rogers, "Lattice vector quantization of image wavelet coefficient vectors using a simplified form of entropy coding," *in Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, Adelaide, Australia, vol. 5, pp. 269-272, 1994.

[22] J. Pan and T. R. Fischer, "Vector quantization-lattice vector quantization of speech LPC coefficients," *in Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, Adelaide, Australia, vol. 1, pp. 513-516, 1994.

[23] M. Xie and J.-P. Adoul, "Embedded algebraic vector quantizers (EAVQ) with application to wideband speech coding," *in Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, Atlanta, USA, vol. 1, pp. 240-243, 1996.

[24] N. Moayeri and D. L. Neuhoff, "Theory of lattice-based fine-coarse vector quantization," *IEEE Transactions on Information Theory*, vol. 37, no. 4, pp. 1072-1084, July 1991.

[25] N. Moayeri, D. L. Neuhoff, and W. E. Stark, "Fine-coarse vector quantization," *IEEE Transactions on Signal Processing*, vol. 39, no. 7, pp. 1503-1515, July 1991.

[26] F. Kuhlmann and J. A. Bucklew, "Piecewise uniform vector quantizers," *IEEE Transactions on Information Theory*, vol. 34, no. 5, pp. 1259-1263, September 1988.

[27] P. F. Swaszek, "Unrestricted multistage vector quantizers," *IEEE Transactions on Information Theory*, vol. 38, no. 3, pp. 1169-1174, May 1992.

[28] T. Eriksson, "Dual-stage vector quantization with dynamic bit allocation," *in Proc. EUSIPCO -94*, Edinburgh, Scotland, vol. 1, pp. 383-386, 1994.

[29] M. V. Eyuboglu and G. D. Forney, "Lattice and trellis quantization with lattice- and trellis-bounded codebooks—High rate theory for memoryless sources," *IEEE Transactions on Information Theory*, vol. 39, no. 1, pp. 46-59, January 1993.

[30] D. G. Jeong and J. D. Gibson, "Uniform and piecewise uniform lattice vector quantization for memoryless Gaussian and Laplacian sources," *IEEE Transactions on Information Theory*, vol. 39, no. 3, pp. 786-804, May 1993.

[31] T. R. Fischer and J. Pan, "Enumeration encoding and decoding algorithms for pyramid cubic lattice and trellis coding," *IEEE Transactions on Information Theory*, vol. 41, no. 6, pp. 2056-2061, November 1995.

[32] P. F. Swaszek, "A vector quantizer for the Laplace source," *IEEE Transactions on Information Theory*, vol. 37, no. 5, pp. 1355-1365, September 1991.

[33] J. H. Conway and N. J. A. Sloane, "A fast encoding method for lattice codes and quantizers," *IEEE Transactions on Information Theory*, vol. 29, no. 4, pp. 820-824, November 1983.

[34] D. G. Jeong and J. D. Gibson, "Lattice vector quantization for image coding," *in Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, Glasgow, Scotland, pp. 1743-1746, 1989.

[35] G. D. Forney, "Multidimensional constellations—Part II: Voronoi constellations," *IEEE Journal on Selected Areas in Communication*, vol. 7, pp. 941-9481989.

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

[37] T. Kohonen, *Self-organizing and associative memory* New York, Springer Verlag, 1984.

[38] K. Zeger, J. Vaisey, and A. Gersho, "Globally optimal vector quantizer design by stochastic relaxation," *IEEE Transactions on Signal Processing*, vol. 40, no. 2, pp. 310-322, February 1992.

[39] N. Ueda and R. Nakano, "A new competitive learning approach based on an equidistortion principle for designing optimal quantizers," *Neural Networks*, vol. 7, no. 8, pp. 1211-12261994.

[40] P. Knagenhjelm, "A recursive design method for robust vector quantization," *in Proc. International Conference on Signal Processing Applications and Technology*, Boston, pp. 948-954, 1992.

[41] J. A. Bucklew, "Companding and random quantization in several dimensions," *IEEE Transactions on Information Theory*, vol. 27, no. 2, pp. 207-211, March 1981.

[42] S. Arya and D. M. Mount, "Algorithms for fast vector quantization," *in Proc. Data Compression Conference*, Snowbird, USA, pp. 381-390, 1993.

[43] J. L. Bentley, “Multidimensional binary search trees used for associative searching,” *Communications of the ACM*, vol. 18, no. 9, pp. 509-517, September 1975.

[44] D. Y. Cheng, A. Gersho, B. Ramamurthi, and Y. Shoham, “Fast search algorithms for vector quantization and pattern matching,” *in Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, San Diego, USA, vol. 1, pp. 9.11.1-9.11.4, 1984.

[45] K. Motoishi and T. Mitsumi, “Fast vector quantization algorithm by using an adaptive search technique,” *in Proc. IEEE International Symposium on Information Theory*, San Diego, USA, pp. 76, 1990.

[46] J. Thyssen and S. D. Hansen, “Using neural networks for vector quantization in low rate speech coders,” *in Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, Minneapolis, USA, vol. 2, pp. 431-434, 1993.

[47] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical recipes in C - the art of scientific computing*, Second ed., Cambridge University Press, 1992.

[48] J. H. Friedman, J. L. Bentley, and R. A. Finkel, “An algorithm for finding best matches in logarithmic expected time,” *ACM Transactions on Mathematical Software*, vol. 3, no. 3, pp. 209-226, September 1977.

[49] D. E. Knuth, *The art of computer programming*, vol. 1, Second ed., Addison-Wesley Publishing Company, 1973.

[50] I. S. Gradshteyn and I. M. Ryzhik, *Table of integrals, series, and products*, 5th ed. San Diego, Academic Press, Inc., 1994.