A Vector Quantizer for the Laplace Source

Peter F. Swaszek, *Member, IEEE*

Abstract—**It is well known that vector quantizers (VQ's) can have substantial gain in performance over scalar quantizers; however, due to their complexity, optimal VQ's are often limited to low bit rate/dimension situations. A low complexity, nearly optimal VQ for the independent Laplace source is presented. Both moderate and high bit rate results are included (greater than one bit per dimension). The VQ is a generalization of Fischer's pyramid VQ and is similar in structure to the unrestricted polar quantizers previously presented for the independent Gaussian source.**

Index Terms—**Vector quantization, source coding, pyramid VQ.**

I. Introduction

A VECTOR quantizer (VQ) is a mapping Q from a continuous input space onto a finite number of points. Typically the input space considered is the k-dimensional Euclidean space $\Re^k$ and the output space is defined as N distinct points (codevectors) in $\Re^k$ (a data rate of $b=(1/k)\log_2 N$ bits per dimension). The input to the VQ is a random vector $\boldsymbol{x}$ with probability density function $f(\boldsymbol{x})$. A k-dimensional, N-level VQ is characterized by the set $\{\mathcal{Q}_i, \hat{\boldsymbol{x}}_i;\ i=1,2,\cdots N\}$ where the $\mathcal{Q}_i$ are the quantization regions (disjoint and covering $\Re^k$) and the $\hat{\boldsymbol{x}}_i$ are their associated codevectors. The VQ operates by mapping all points of $\mathcal{Q}_i$ onto $\hat{\boldsymbol{x}}_i$. As one solution to the source encoding problem, the performance advantages of vector quantizers have been known for over 25 years [33]. Much relevant work in the area has appeared since. Most of the available *design* work on VQ's is in the low resolution region [7], [12], [18]; either this is the region of interest (e.g., image coding) or the design algorithms are impractical at higher rates. In the high rate region the previous research is often nonconstructive [6], [14], [33] (bounds on performance, etc.) or dependent upon a particular source model [9], [13], [27], [30].

As a performance criterion for this discussion, consider the mean-squared error per dimension (mse) which has experienced wide application due to its tractability and interpretation as quantization noise power. With the above notation, the mse is

$$\text{mse} = \frac{1}{k}\int_{\Re^k} |\boldsymbol{x}-Q(\boldsymbol{x})|^2 f(\boldsymbol{x})\,d\boldsymbol{x} = \frac{1}{k}\sum_{i=1}^{N}\int_{\mathcal{Q}_i} |\boldsymbol{x}-\hat{\boldsymbol{x}}_i|^2 f(\boldsymbol{x})\,d\boldsymbol{x}. \quad (1)$$

The *optimum* k-dimensional, N region VQ has been shown [33] for large N to have mse performance

$$\text{mse}_{\text{opt}} \approx \frac{G_k}{N^{2/k}}\left\{\int_{\Re^k} f(\boldsymbol{x})^{k/(k+2)}\,d\boldsymbol{x}\right\}^{(k+2)/k}, \quad (2)$$

where G_k is a constant dependent only upon the dimension [9], [14], typically thought of as the coefficient of quantization for a uniform source. This approximate mse result helps to predict the gain in performance of VQ's over scalar quantizers. It can be shown that performance increases with an increase in dimension, even if the source variables are independent [20]. Unfortunately this result is not constructive; hence, researchers have developed methods for constructing VQ's, with (hopefully) near optimum performance.

One approach for constructing VQ's, which has been successful for spherically symmetric source models, is to transform the source variables to (spherical) coordinates that describe contours of constant probability and employ product code quantizers on this new coordinate set [1], [4], [5], [24], [29]. To improve performance the product code quantizers were modified to allow variable resolution dependent upon the earlier quantizations. These *unrestricted VQ's* [30], [32] asymptotically achieved nearly optimum performance. In this paper, we describe an unrestricted VQ appropriate for the independent and identically distributed (i.i.d.) Laplace source. Assuming zero means and unit variances, the source vector $\boldsymbol{x}$ with elements $x_1,\cdots,x_k$ has probability density function

$$f(\boldsymbol{x}) = \prod_{i=1}^{k}\frac{1}{\sqrt{2}}e^{-\sqrt{2}|x_i|} \quad (3)$$

and is commonly used to model differential speech [23] and the discrete cosine transform of image data [25]. For this source, the mse gain of vector quantization over scalar quantization can be up to 2.44 dB in dimension two, 3.33 dB in dimension three, growing in the limit of large dimension to 7.16 dB. These numbers are asymptotic gains for high resolution quantization. Similar results

Manuscript received April 11, 1990; revised December 11, 1990. This work was supported in part by the National Science Foundation under Grant ECS-8521844. This work was presented in part at the MITRE Corp.'s Quantization and Analog-to-Digital Conversion Workshop, Bedford, MA, July 1986, and in part at the IEEE International Symposium on Information Theory, San Diego, CA, January 14–19, 1990.

The author is with the Department of Electrical Engineering, Kelley Hall, University of Rhode Island, Kingston, RI 02881. e-mail: swaszek@quahog.uri.edu.

IEEE Log Number 9101607.

for low bit rates can be computed from the rate distortion function [22].

The structure of this unrestricted VQ for the Laplace source is based on concentric pyramids (versus concentric spheres in [30], [32]) and is directly related to Fischer's pyramid VQ [13]. There are several key differences between our approach and Fischer's.

- Derivation of the performance of the pyramid VQ's in [13] follows rate distortion theory arguments, letting the dimension of the system increase until pyramid hardening occurs. In contrast, our derivation is based on quantization theory and is asymptotic in the bit rate, not dimension. Our results are valid for any number of dimensions.
- Although the derivation of [13] assumes one pyramid, the example pyramid VQ's are product code form; a gain scalar is quantized followed by a quantization of the location on the pyramid. Our VQ begins with concentric pyramids; we derive results for both the product code arrangement as well as the more versatile unrestricted format.
- Although not stated directly in [13], the pyramid VQ's are based on translates of the A_n lattice [9], [15] (described below) as a VQ subsystem. The presentation herein permits the use of any lattice VQ.

We believe this work to be a useful complement to that of [13].

This paper begins with a description of the VQ in k dimensions. Next, an analysis of performance is presented with results for both the product code pyramid VQ as well as the unrestricted version. This analysis, although asymptotic in nature, helps to demonstrate the performance advantages of our VQ. Details of the analysis are relegated to the Appendix. Implementation issues of the VQ are discussed next with emphasis on the A_n lattice case. Nonasymptotic results are then considered. In particular, we present an approximate design algorithm for finite bit rate and demonstrate the usefulness of this VQ through several example designs with Monte Carlo simulations of performance.

II. The VQ in k Dimensions

The source probability density function in (3) has contours of constant probability that are scaled versions of the cross polytope β_k [11, pp. 120–121] centered on the origin. This polytope has $2k$ vertices, two on each of the original Euclidean axes, and 2^k faces, each a regular simplex, α_{k-1}, of dimension $k-1$. For dimension two, this contour is a diamond; in dimension three, a regular octahedron.

Since the VQ structure depends upon a change of coordinates to those of constant probability, we consider the gain g and $k-1$ location variables u_j, $j=1,\cdots,k-1$, defined by

$$g=\frac{1}{\sqrt{k}}\sum_{i=1}^{k}|x_i|, \qquad u_j=\frac{1}{\sqrt{j(j+1)}}\left(\sum_{i=1}^{j}|x_i|-j|x_{j+1}|\right).$$

Note that to be able to undo this transformation we need the signs of each x_i (equivalent to k bits). This transformation (sometimes called Helmert's [17, pp. 12–14]) is orthogonal. The coordinate g specifies the size of the polytope and the u_j specifies the location on one of its faces. For a fixed value of g the polytope has edges of length $g\sqrt{2k}$. Further, each face, a simplex, has volume (area in dimension k)

$$\text{vol}\{\alpha_{k-1}(g)\}=\frac{k^{k/2}}{(k-1)!}g^{k-1}.$$

Changing variables and letting $\boldsymbol{u}$ represent the vector $[u_1,\cdots,u_{k-1}]$, the density function for the source becomes

$$f(g,\boldsymbol{u})=2^{k/2}e^{-\sqrt{2k}g},$$

and is not explicitly a function of $\boldsymbol{u}$. In other words, conditioned on the value of g, $\boldsymbol{u}$ is *uniformly distributed* on the simplex $\alpha_{k-1}(g)$. For $k=2$ this simplex is a line interval, for $k=3$ an equilateral triangle, for $k=4$ a regular tetrahedron, etc. Integrating over the simplex, the marginal density of g is

$$f(g)=\frac{(2k)^{k/2}}{(k-1)!}g^{k-1}e^{-\sqrt{2k}g}.$$

The *unrestricted vector quantizer* (UVQ) is implemented as follows:

1) transform from $\boldsymbol{x}$ to g and $\boldsymbol{u}$, save the signs of the x_i;
2) scalar quantize g to N_g levels, call the result $\hat{g}_i$;
3) project $\boldsymbol{u}$ onto $\alpha_{k-1}(\hat{g}_i)$ (this is necessary only if $g>\hat{g}_i$);
4) enter a lookup table based upon the result of Step 2) for $N_u(\hat{g}_i)$, the number of regions for (resolution of) the $\boldsymbol{u}$ quantizer;
5) quantize $\boldsymbol{u}$ to the resolution found in Step 4) using a lattice vector quantizer of dimension $k-1$ on the simplex $\alpha_{k-1}(\hat{g}_i)$ (this simplex has edge length $\hat{g}_i\sqrt{2k}$);
6) transmit and receive an index based on Steps 2) and 5) over the channel; and
7) retransform to $\hat{\boldsymbol{x}}$.

For example, Fig. 1 shows the $N=16$ UVQ in dimension two. Note that g is quantized to $N_g=2$ levels. The u variable, a scalar, falls on the real line between $-g$ and $+g$. Our simplex, an interval of length $2\hat{g}$, is the segment from $-\hat{g}$ to $+\hat{g}$. The lattice quantizer for u is a uniform scalar quantizer with either two or four levels. We set this uniform quantizer so that $-\hat{g}$ and $+\hat{g}$ are the extreme representation points. Note that for $k=2$ we have only one choice for the lattice (intervals on $\Re^1$) while for $k>2$ we have many choices. Below we present results for any choice of the lattice. Specifics for the A_n lattices [9], [15] are described.

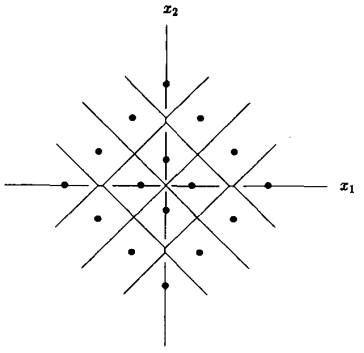

Fig. 1. Dimension two, $N = 16$ UVQ.

III. Asymptotic Analysis

The goal of asymptotic (high bit rate, $N \to \infty$) quantization theory is first to predict performance, and second, for use in design. For example, Bennett's integral [3] describes the minimum mse for a scalar quantizer; the compressor function can then be used to provide approximate quantizer parameters.

Let $S_{k-1}(\hat{g})$ be a typical quantization region of the $k-1$ dimensional lattice VQ for $\boldsymbol{u}$ on $\alpha_{k-1}(\hat{g})$ in Step 5) of our VQ algorithm. (Actually the boundary of $\alpha_{k-1}(\hat{g})$ distorts the shapes of some of the VQ regions; for our analysis we will ignore these effects). Let U_{k-1} be the inertia of a normalized (unit volume) version of this region (equivalent to $U(S_{k-1}(\hat{g}))$ using the notation of [9]). It is shown in the appendix that the mse for the UVQ with an i.i.d. Laplace input can be approximated by

$$\text{mse} \approx \frac{1}{k}\int_0^\infty \left[(g-\hat{g})^2 \frac{k^{k/2} g^{k-1}}{(k-1)!} + \frac{N_u(\hat{g}) U_{k-1}}{2^k} \cdot \left[\text{vol}\{S_{k-1}(\hat{g})\}\right]^{(k+1)/(k-1)} \right] 2^{k/2} e^{-\sqrt{2k}g}\, du\, dg.$$

Details of an asymptotic analysis appear in the Appendix. Specifically, we replace the rate allocation sequence $N_u(\hat{g})$ by an allocation function $N_u(g)$ and consider a compandor implementation for the g quantizer with compressor function h, expandor h^{-1}, and N_g levels in the scalar uniform quantizer. A Bennett-like expression for the mse is developed, which is dependent upon the choice of the functions $N_u(g)$ and $h(g)$ and the parameters N_g and U_{k-1}. This expression is

$$\text{mse} \approx \frac{(2k)^{k/2}}{12k!N_g^2}\int_0^\infty \frac{g^{k-1}e^{-\sqrt{2k}g}}{|h'(g)|^2}\, dg + \frac{(k-1)(2k)^{k/2}U_{k-1}}{k!}\left(\frac{2^k k^{k/2}}{(k-1)!}\right)^{2/(k-1)} \cdot \int_0^\infty g^{k+1} N_u^{-2/(k-1)} e^{-\sqrt{2k}g}\, dg. \quad (4)$$

From this point we can follow two directions to optimize this expression: force $N_u(g)$ to be a constant (Fischer's pyramid vector quantizer) or consider the unrestricted format (UVQ).

The Pyramid VQ: The pyramid vector quantizer is based on constant resolution for each pyramid

$$N_u(g) = \frac{N}{N_g}.$$

For fixed values of k and N the mse is then only a function of the gain compressor, $h(g)$, the gain resolution, N_g, and the uniform vector quantizer for $\boldsymbol{u}$. Optimization over N_g (equating the derivative to zero) and $h(g)$ (the calculus of variations or Hölder's inequality) yields

$$\text{mse}_{\text{Pyramid VQ}} \approx 6[(k+1)kU_{k-1}]^{(k-1)/k} 3^{2/k} \cdot \left(\frac{\Gamma\left(\frac{k+2}{3}\right)}{(k-1)!}\right)^{3/k} \left(\frac{1}{12}\right)^{1/k} N^{-2/k}.$$

This mse result will be compared to other VQ's. We note that this result is not exactly applicable to Fischer's pyramid VQ; in [13] the gain quantizer is a Gaussian quantizer, different from our best pyramid VQ compressor. However, for larger k the compressors converge (a central limit theorem argument) and we expect little difference in the two results. Also in [13] the quantization codevectors on the pyramid are forced to satisfy the relation

$$\sum_{i=1}^{k} |x_i| = m,$$

with m and each x_i being integers. The lattice A_{k-1} is defined as that subset of the dimension k integer lattice $\boldsymbol{Z}^k$ whose coefficients sum to zero [9], [15]

$$\sum_{i=1}^{k} x_i = 0$$

(i.e., those $\boldsymbol{Z}^k$ lattice points that fall on a hyperplane slicing through the origin). Fischer's VQ codevectors then are a translate of this lattice by m units normal to this hyperplane. With this realization, U_{k-1} for the pyramid VQ is the normalized inertia for the A_{k-1} lattice [9]

$$U_{k-1} = (k-1)\sqrt{k}\left(\frac{1}{12} + \frac{1}{6k}\right).$$

It can be shown that the A_{k-1} lattice has other properties that will be especially convenient for implementation. For example, one can scale the lattice so that lattice points fall directly on the vertices of the simplex $\alpha_{k-1}(\hat{g})$. Further, a simple implementation for the UVQ exists (and is described later).

The Unrestricted VQ: The total number of representation points in $\mathfrak{R}^k$ is a fixed constant N. As a function of

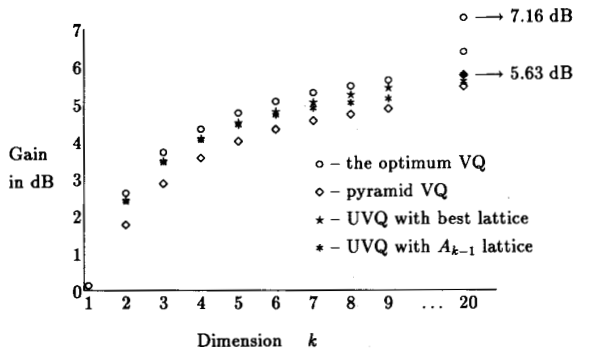

Fig. 2. Asymptotic gains in mse over scalar quantization.

the choices of N_g and $N_u(\hat{g}_i)$ this can be written as

$$\sum_{i=1}^{N_g} N_u(\hat{g}_i) = N. \tag{5}$$

With this constraint we can minimize the mse over the choice of the levels allocation function $N_u(g)$ (using a Lagrange multiplier and the calculus of variations), the g quantizer's resolution N_g (equate the derivative to zero) and compressor $h(g)$ (apply Hölder's inequality). The main result is

$$\text{mse}_{\text{UVQ}} \approx 2\left(\frac{k+2}{k}\right)^{k+2} U_{k-1}^{(k-1)/k}\left(\frac{1}{12}\right)^{1/k} N^{-2/k}. \tag{6}$$

For comparison, the minimum mse from (2) for this source is

$$\text{mse}_{\text{opt}} \approx 2\left(\frac{k+2}{k}\right)^{k+2} G_k N^{-2/k}.$$

We note that these last two mse expressions are of very similar forms; the difference is the term due to the inertia of the basic quantization region. In the optimum VQ, this inertia is G_k; for the UVQ our basic region is a cross product of an interval for g (yielding the $(\frac{1}{12})^{1/k}$ term) and a dimension $k-1$ region for $\boldsymbol{u}$ (the $U_{k-1}^{(k-1)/k}$ term). To compare all three performance expressions (pyramid VQ, UVQ, and optimum VQ) we plot the gain in mse performance of each over scalar quantization. For the same resolution a scalar quantizer has performance

$$\text{mse}_{\text{scalar}} \approx 4.5N^{-2/k},$$

so Fig. 2 shows versus dimension k the gain in dB

$$10\log_{10}\frac{\text{mse}_{\text{scalar}}}{\text{mse}_{\text{VQ}}} = 10\log_{10}\frac{4.5}{\text{mse}_{\text{VQ}}}.$$

For the unrestricted VQ we present the gains using the A_{k-1} lattice and Conway and Sloane's [9] best lattice ($U_{k-1} = G_{k-1}$). In the limit of large k the optimum VQ's performance gain approaches 7.16 dB while the UVQ , with the A_{k-1} lattice, and the pyramid VQ approach 5.63 dB (the 1.5 dB loss is due to poor performance of that lattice). For small dimension, the UVQ's performance is better than that of the pyramid VQ. Further, the UVQ's performance varies little with lattice choice. For higher dimension one might want to implement other lattice VQ's for $\boldsymbol{u}$.

IV. Design and Implementation Issues

To design and implement the UVQ's we must further address three issues: projection onto $\alpha_{k-1}(\hat{g})$ as in Step 3), implementation of the lattice VQ for $\boldsymbol{u}$, and selection of the VQ's parameters for finite N. Since the A_n lattice has excellent performance for small dimension, we limit much of our discussion to this case only.

If we desire to use a general lattice VQ for $\boldsymbol{u}$ on $\alpha_{k-1}(\hat{g})$ we must first project $\boldsymbol{u}$ onto $\alpha_{k-1}(\hat{g})$ if $g > \hat{g}$. The reason is that if our input vector falls outside the simplex, the uniform VQ algorithm [10] might map $\boldsymbol{u}$ onto a quantization codevector also outside the simplex and, hence, not in our codebook. This projection can be achieved by sequentially projecting $\boldsymbol{u}$ onto the lower dimensional faces of $\alpha_{k-1}(\hat{g})$. For the A_{k-1} lattice this step is subsumed into the VQ algorithm described next.

As previously noted, a translate of the A_{k-1} lattice is characterized by points $\boldsymbol{y} = [y_1, \cdots, y_k]$ with

$$\boldsymbol{y} \in \boldsymbol{Z}^k \qquad \text{such that } \sum_{i=1}^{k} y_i = m,$$

where m is an integer. We notice that, if one scales the VQ input $\boldsymbol{x}$ by $m/\hat{g}$, then our simplex $\alpha_{k-1}(\hat{g})$ for $\boldsymbol{u}$ lies within the hyperplane of this lattice. The k vertices of the simplex are lattice points and have coordinate representations which are the k permutations of the vector $[m, 0, \cdots, 0]$. Further, with the scaling by m, any one edge of this simplex contains exactly $m+1$ evenly spaced lattice points. Realizing this we can develop a relationship between m and the total number of VQ points on β_k. From [11, pp. 120–121] there are $2^{d+1}\binom{k}{d+1}$ d-dimensional faces on the polytope β_k. The number of lattice vectors *interior* to each face is $\binom{m-1}{d}$ for $m > d$ (zero otherwise). Combining these, the total number of vectors on β_k as a function of k and m is

$$N_u = \sum_{d=0}^{k-1}\left(\begin{matrix}\text{number}\\ \text{of faces}\end{matrix}\right)\times\left(\begin{matrix}\text{number of}\\ \text{interior points}\end{matrix}\right)$$

$$= \sum_{d=0,\, d<m}^{k-1} 2^{d+1}\binom{k}{d+1}\binom{m-1}{d}. \tag{7}$$

The overall VQ algorithm, then, is a slight modification of that in [13] in that we select m based upon the result of the gain quantization:

1) let $g = \sum_{i=1}^{k}|x_i|$, quantize g to $\hat{g}$, look up $m(\hat{g})$;
2) replace each x_i by

$$|x_i| + \frac{\hat{g} - g}{\sqrt{k}},$$

a perpendicular projection onto the A_{k-1} lattice hyperplane at distance $\hat{g}$ from the origin;
3) let $y_i = x_i \times m(\hat{g})/\hat{g}$;

4) round each y_i to the nearest integer in the set $\{0,1,2,\cdots,m(\hat{g})\}$, compute $T=\sum_{i=1}^{k} y_i$;
5) IF $T=m$, THEN GO TO Step 6),
IF $T>m$, round an additional $T-m(\hat{g})$ y_i's down by 1,
IF $T<m$, round an additional $m(\hat{g})-T$ y_is up by 1;
6) restore the sign of each coordinate, i.e., $y_i=\mathrm{sgn}(x_i)\times y_i$;
7) $\hat{x}_i=y_i\times\hat{g}/m(\hat{g})$.

Between Steps 6) and 7) of the VQ algorithm, it is common to encode and decode the value for transmission/storage over a channel. We require then a method of encoding the y_i onto an integer between 1 and $N=2^{kb}$. Our approach, described next, is a minor variation of enumeration encoding given in [13].

A. Encoding

The problem is: Given $\mathbf{y}=[y_1,y_2,\cdots,y_k]$ with

$$\sum_{i=1}^{k}|y_i|=m(\hat{g}),$$

find a unique index z, $z\in\{1,2,\cdots,N\}$. Assume that we assign indexes to the VQ vectors sequentially over the concentric cross polytopes $\hat{g}\times\beta_k$ starting with the innermost shell. The overall index of a particular codevector can then be written as an offset O_0 due to the points on all interior shells plus the position index, P_0, on the current shell:

$$\text{index of } \mathbf{y}=O_0(\hat{g})+P_0(\mathbf{y}).$$

Note that this offset for a codevector on the jth shell of magnitude $\hat{g}_j$ is given by

$$O_0(\hat{g}_j)=\sum_{i=1}^{j-1}N_u(\hat{g}_i),$$

and that the position, P_0, on the current shell is an element of $\{1,2,\cdots,N_u(\hat{g}_j)\}$.

The position on the shell can be written as the sum of an offset O_1 due to all codevectors with smaller values of y_1 plus the position P_1 of the point of interest within those codevectors on the shell with the given y_1 value:

$$P_0(\hat{g})=O_1(m(\hat{g}),y_1)+P_1(\mathbf{y}).$$

This second offset can be found as

$$O_1=\sum_{j=-m(\hat{g})}^{y_1-1}\left\{\text{number of ways that } \sum_{i=2}^{k}|y_i|=m(\hat{g})-|j|\right\}.$$

Similarly, we can find P_1 as an offset of all codevectors on the shell with the given y_1 value and y_2 less than but not equal to the given value, plus another position. Continuing, the overall index for any point can be written as

$$\text{index}=O_0+\sum_{i=1}^{k}O_i,$$

TABLE I
LOOKUP TABLES FOR THE FOUR-DIMENSIONAL, RATE 3 ($N=4041$) UVQ: THE GAIN QUANTIZER'S PARAMETERS AND THE $\tilde{O}_i$ USED IN THE ENCODING STEP

i	$\hat{g}_i$	g_i	$m(\hat{g}_i)$	$O_0(\hat{g}_i)$	j	$\tilde{O}_1(j)$	$\tilde{O}_2(j)$	$\tilde{O}_3(j)$
1	0.	0.12	0	0	0	1	1	1
2	0.18	0.36	2	1	1	7	5	3
3	0.51	0.71	4	33	2	25	13	5
4	0.89	1.1	6	225	3	63	25	7
5	1.3	1.6	6	833	4	129	41	9
6	1.8	2.2	7	1441	5	231	61	11
7	2.5	3.1	7	2393	6	377	85	13
8	3.4	4.5	6	3345	7	575	113	15
9	5.1	∞	3	3953				

where, for $i=1,2,\cdots,k-1$,

$$O_i=\sum_{j=-(m(\hat{g})-\sum_{l=1}^{i-1}|y_l|)}^{y_i-1}\left\{\text{number of ways that } \sum_{p=i+1}^{k}|y_p|=m(\hat{g})-\sum_{q=1}^{i-1}|y_q|-|j|\right\}$$

and

$$O_k=\begin{cases}1, & y_k\le 0,\\ 2, & y_k>0.\end{cases}$$

A simpler way to compute these offsets is as follows. First, define the functions $\tilde{O}_i(j)$ $(i=1,2,\cdots,k-1,\ j=0,1,2,3,\cdots,\max\{m(\hat{g})\})$ by

$$\tilde{O}_i(j)=\text{number of ways that } \sum_{p=1}^{k-i}|x_p|\le j,$$

each x_p an integer.

Notice that we can define these recursively as

$$\tilde{O}_i(0)=1,$$

$$\tilde{O}_{k-1}(j)=1+2j, \text{ and } \tilde{O}_{n-1}(j)=\tilde{O}_n(j)+2\sum_{p=1}^{j-1}\tilde{O}_n(p).$$

The original offsets can then be found as

$$O_i(y_1,\cdots,y_k)=\begin{cases}0, & y_i\le 0 \text{ and } b_i=0,\\ \tilde{O}_i(b_i-1), & y_i\le 0 \text{ and } b_i>0,\\ \left(\tilde{O}_{i-1}(b_{i-1})-\tilde{O}_{i-1}(b_{i-1}-1)\right)-\tilde{O}_i(b_i), & y_i>0,\end{cases}$$

where $b_i=m(\hat{g})-\sum_{p=1}^{i}|y_p|$ $(b_i\in\{0,1,\cdots,m(\hat{g})\})$ and $\tilde{O}_0(\cdot)-\tilde{O}_0(\cdot)$ is interpreted as the number of points on the jth shell. For simplicity of implementation of both the encoder and decoder, the values of $\tilde{O}_i$ should be tabulated.

As an example, we consider the dimension four, rate 3 ($N=4041$) UVQ designed using the approach of the next section. This UVQ has 9 concentric shells. The offsets O_0, the g quantizer's parameters, and the $\tilde{O}_i(\cdot)$ appear in Table I. The encoding of the point $y_1=2$, $y_2=-2$, $y_3=0$, $y_4=3$ $(m=7)$ on the sixth shell starts with

$O_0(6) = 1441$. Next,

$$b_1 = 7-(2) = 5 \quad \rightarrow O_1 = (2393-1441)-231 = 721,$$
$$b_2 = 7-(2+2) = 3 \quad \rightarrow O_2 = 13,$$
$$b_3 = 7-(2+2+0) = 3 \quad \rightarrow O_3 = 5,$$

and since $y_4 > 0$, $O_4 = 2$. Summing, the encoded index is $1441+721+13+5+2 = 2182$.

Decoding is the reverse operation: given the index, find that offset nearest to but less than the index. Subtract this value and search over the next offset. For example, to decode the previous example of index equal to 2182, we start by selecting that value of $O_0 < 2182$ to determine that our point is on the sixth shell. This gives us the value $m(\hat{g}) = 7$ from the first lookup table. To continue, we subtract the shell offset from the index

$$P_0 = \text{index} - O_0 = 2182 - 1441 = 741.$$

We then solve for O_1. From the lookup table, 741 is greater than any value of $\tilde{O}_1(j)$ for $j \le m(\hat{g})$; hence, y_1 must be greater than zero. In that case, the position must exceed 952 (the number of points on the sixth shell) minus the $\tilde{O}_1(b_1)$ entry; i.e., $b_1 = 5$ ($741 > 952-231 = 721$) and $y_1 = m(\hat{g}) - b_1 = 2$. Continuing, the remaining position is now $P_1 = P_0 - O_1 = 741-721 = 20$. We can find $b_2 = 3$ from the table (the $j = 2$ entry); hence, $y_2 = -2$. Now, $P_2 = P_1 - O_2 = 20-13 = 7$, so $b_3 = 3$ ($j = 2$) and $y_3 = 0$. Finally, the remaining index is $7-5 = 2$ so $y_4 > 0$ and equals $m(\hat{g}) - |y_1| - |y_2| - |y_3| = 7-(2+2) = 3$.

B. Complexity

Besides designing for the best performance, we must consider implementation details of the VQ. Any VQ can be divided into two operations, an encoder and a decoder, a model useful when comparing complexity of implementation of several schemes. Since this is an important issue in VQ design, we compare the computation and storage requirements of the implementation of scalar, unstructured (locally optimum) vector quantizers, and our UVQ for the length k vector input.

In the scalar case, the same scalar quantizer is employed k times. At a rate of b bits per sample, this scalar quantizer has 2^b intervals as its quantization regions. Encoding a scalar input is accomplished by comparing the input to the interval endpoints in a binary tree fashion. Such an operation requires b comparisons and b table lookups (of the interval endpoints) per input scalar. In total, the implementation of the scalar quantizer's encoder for a vector of length k requires $2^b - 1$ storage locations (for the endpoints) and $2kb$ scalar operations (comparisons and lookups). For the decoder, k lookups in a table of the 2^b scalar quantizer's output values are required. These counts of scalar operations and scalar storage locations are summarized in Table II.

For a typical unstructured VQ, b bits per sample yields a total of kb bits or 2^{kb} outputs. Since the vectors are not ordered as in the scalar case the encoder is often implemented by a straightforward technique, called a full search, which computes and compares the distances to each output. Thus the encoder requires a table of the outputs and the computation of the distance to each. The outputs, each filling k locations, require a memory with $k2^{kb}$ storage locations. Each distance computation involves k subtractions, k multiplies (squaring), and $k-1$ additions. Finally, $2^{kb}-1$ scalar comparisons yield the closest output. For the decoder, one lookup in a table of the 2^{kb} codevectors is required (see Table II). This exponential growth (in both k and b) motivated research into lower complexity search algorithms [2], [8], tree search VQ's [7], multistage VQ's [16], and product code quantizers [26].

TABLE II
COMPARISON OF IMPLEMENTATION COMPLEXITY OF SEVERAL VQ'S

Type of VQ	Number of Operations	Amount of Memory
Scalar quant.	$O(kb)$	$O(2^b)$
Unstructured VQ (full search)	$O(k2^{kb})$	$O(k2^{kb})$
UVQ	$O(k+2^b)$	$O(k2^b)$

The complexity of the UVQ's implementation is more involved to compute. The full algorithm for the A_{k-1} lattice UVQ (including the encoder and decoder) previously described in this section requires approximately $16k + 2\log_2 N_g + \max\{m(\hat{q})\}$ scalar operations (additions, multiplications, comparisons, table lookups, etc.) and $13N_g + \max\{m(\hat{g})\}$ scalar storage locations. Experimental observation that both N_g and the maximum $m(\hat{g})$ are proportional to 2^b yields the estimates listed in Table II.

C. Finite N

The asymptotic results already presented demonstrate the performance gain of the UVQ's. For finite N, the relevant question is: *How do we design the UVQ*?

Previously [31], [34] we have presented example UVQ's for the dimension two Laplace source for finite bit rates. The resulting quantization patterns (on the bivariate plane) closely resemble the optimum VQ's found using the LBG algorithm [12]. In this case the mse expression for the VQ can be written explicitly in terms of the quantizer parameters and optimized by a Lloyd method I approach [19]. Although nonlinear and coupled, the partial derivative expressions can be solved numerically [34]. The gains in performance over scalar quantization [21], [23] for 2, 3, and 4 bits per dimension are 0.68, 1.47, and 1.78 dB, respectively. Unfortunately, for $k > 2$, the geometry of the quantization regions becomes too complex to allow a simple, exact, finite N design algorithm. Clearly, one solution is to employ the asymptotic results directly; i.e., given k and $N = 2^{kb}$:

- evaluate N_g from (A7);
- inverse sample the compressor h of (A9) at N_g uniformly spaced points for the $\hat{g}_i$; select the g thresholds as midpoints of the $\hat{g}_i$,
- evaluate the $N_u(\hat{g}_i)$ using (A8) and round to allowable values of N_u that satisfy the constraint in (7).

TABLE III
DATA FOR A $N = 57$, THREE DIMENSIONAL UVQ

i	$\hat{g}_i$	$N_u(\hat{g})$	N_u	m	g_i	$\hat{g}_i$
1	0.000	0	1	0		0.000
2	1.043	31	18	2	0.3789	0.6973
3	3.285	33	38	3	1.337	1.793

Unfortunately, this method results in poor performance unless the per dimension bit rate is high (e.g., $b \geq 8$). The major reason for this is that the asymptotic analysis ignores the effects of regions on the boundaries of the faces of β_k. Specifically, the previous analysis assumes that all of the regions are of equal size and contribute equally to the error while for small N these boundary regions are larger than the rest and dominate the error expression.

A finite N algorithm is developed in the Appendix that does provide good VQ performance for lower bit rates. If one makes an initial choice for N_g and the $N_u(\hat{g})$ (using the asymptotic results), the performance is then a function of only the gain quantizer's parameters. Taking derivatives, the result is a pair of Lloyd/Max equations [19] for the g quantizer:

$$\hat{g}_i = \frac{\int_{g_i}^{g_{i+1}} g^k e^{-\sqrt{2k}g}\, dg}{(1+\delta_i)\int_{g_i}^{g_{i+1}} g^{k-1} e^{-\sqrt{2k}g}\, dg}$$

and

$$g_i = \frac{\hat{g}_i^2(1+\delta_i) - \hat{g}_{i-1}^2(1+\delta_{i-1})}{2(\hat{g}_i - \hat{g}_{i-1})},$$

where

$$\delta_i = \frac{N_u(\hat{g}_i)\, k!\sqrt{2}\, \gamma_{k-1}}{2^{k/2}\sqrt{k}}$$

and γ_{k-1} is constant dependent upon the particular lattice VQ employed. For the A_{k-1} lattice

$$\gamma_{k-1} = \frac{(k-1)\sqrt{k}}{2^{(k+1)/2} m^{k+1}} \left(\frac{1}{12} + \frac{1}{6k} \right).$$

These expressions assume that the innermost pyramid has multiple quantization points. If we consider a single small region centered at the origin ($\hat{g}_1 = 0$), the expression for g_2 becomes

$$g_2 = \frac{k+1}{k-1} \hat{g}_2 \left(-1 + \sqrt{\frac{k(2+\delta_2) - \delta_2}{k+1}} \right).$$

For example consider the results for $k = 3$ and $N = 64$ ($b = 2$) in Table III. Evaluating the asymptotic expression (A7) and rounding to the nearest integer yields $N_g = 3$. We will consider the case of a representation vector at the origin. Sampling h^{-1} at 0, 2/5, and 4/5 yields the $\hat{g}_i$ (column 2). Next, we sample $N_u(g)$ at the $\hat{g}_i$ and round to integers that sum to 64 to get approximations to the $N_u(\hat{g})$ (column 3). We then select N_u for integer m (columns 4 and 5). We note that the total is $N = 57$ regions. Unfortunately, this VQ has poor performance; its mse is larger than that of scalar quantization. To improve performance we need to reselect the g quantizer and, possibly, vary the $N_u(i)$. Keeping the $N_u(i)$ fixed and iterating over g_i and $\hat{g}_i$ with the expressions above yields the data in columns 6 and 7. The performance of this VQ (by Monte Carlo simulation) is 1.13 dB better than that of scalar quantization. Results of several other example UVQ's for low dimension and moderate bit rates appear in Table IV.

TABLE IV
SIMULATION RESULTS OF PERFORMANCE (GAIN IN dB OVER SCALAR QUANTIZATION) FOR VARIOUS LOW DIMENSION (k), MODERATE BIT RATE (b) LAPLACE UVQ'S

k	b	2^{bk}	Actual N	N_g	SNR Gain
2	2	16	16	2	0.68 dB
3	2	64	57	3	1.45 dB
4	2	256	209	4	1.58 dB
5	2	1024	950	3	1.93 dB
2	3	64	64	5	1.47 dB
3	3	512	511	8	2.37 dB
4	3	4096	4041	9	2.53 dB
5	3	32768	31050	9	2.73 dB
2	4	256	256	11	1.78 dB
3	4	4096	4096	14	2.85 dB
4	4	65536	65504	17	3.12 dB
5	4	1048576	1031381	19	3.39 dB

V. CONCLUSION

This paper has presented asymptotic (high bit rate) and finite N design and performance analyses for an unrestricted VQ for the independent Laplace source. For the restricted form (the pyramid VQ), this paper provides further implementational information and low dimension analytical results. The introduction of the unrestricted format allows us to use lower dimension yet achieve similar gains in mse performance.

Normally, when discussing VQ schemes, the question of implementational complexity is discussed. It was shown that the UVQ's complexity (gain quantization, $\boldsymbol{u}$ quantization, encoding and decoding) is significantly smaller than that of the locally optimum VQ's typically designed.

APPENDIX

This appendix presents an analysis of the unrestricted VQ based on Helmert's transformation for an i.i.d. Laplace source. Both asymptotic and finite N analyses are presented along with the restriction to a product code pyramid vector quantizer.

For a dimension k source $\boldsymbol{x}$ with probability density function $f(\boldsymbol{x})$, the mse due to quantization is

$$\text{mse} = \frac{1}{k} \int_{\Re^k} |\boldsymbol{x} - Q(\boldsymbol{x})|^2 f(\boldsymbol{x})\, d\boldsymbol{x}$$

$$= \frac{1}{k} \sum_{i=1}^{N} \int_{\mathcal{Q}_i} |\boldsymbol{x} - \hat{\boldsymbol{x}}_i|^2 f(\boldsymbol{x})\, d\boldsymbol{x}.$$

Here, we are concerned with a source modeled by the

i.i.d. Laplace probability density function

$$f(\boldsymbol{x}) = \prod_{i=1}^{k} \frac{1}{\sqrt{2}} e^{-\sqrt{2}|x_i|}$$

and quantization of the variables from Helmert's transformation

$$g = \frac{1}{\sqrt{k}} \sum_{i=1}^{k} |x_i|,$$

$$u_j = \frac{1}{\sqrt{j(j+1)}} \left(\sum_{i=1}^{j} |x_i| - j|x_{j+1}| \right), \qquad j = 1, \cdots, k-1.$$

We call g the *gain* coordinate and $\boldsymbol{u} = [u_1, \cdots, u_{k-1}]$ the *location* vector on the polytope β_k. Changing coordinates we note that Helmert's transformation is orthogonal; the mse expression becomes

$$\text{mse} = \frac{1}{k} \int_0^{\infty} \int_{\alpha_{k-1}(g)} \left[(g - \hat{g})^2 + \sum_{j=1}^{k-1} (u_j - \hat{u}_j)^2 \right] \cdot 2^{k/2} e^{-\sqrt{2k}g} \, d\boldsymbol{u} \, dg,$$

with $\alpha_{k-1}(g)$ a regular simplex with edge length $\sqrt{2k}\,g$.

To simplify this mse expression, we begin with the inner integration over the u_j,

$$\int_{\alpha_{k-1}(g)} \left[(g - \hat{g})^2 + \sum_{j=1}^{k-1} (u_j - \hat{u}_j)^2 \right] d\boldsymbol{u}. \qquad \text{(A.1)}$$

The first term of this integral can be integrated directly,

$$\int_{\alpha_{k-1}(g)} (g - \hat{g})^2 \, d\boldsymbol{u} = (g - \hat{g})^2 \times \text{vol}\{\alpha_{k-1}(g)\}.$$

Since $\boldsymbol{u}$ is uniformly distributed over $\alpha_{k-1}(g)$, the second term of (A1) is proportional to the error power due to the $\boldsymbol{u}$ quantization; equivalently, it is the total inertia of the $\boldsymbol{u}$ quantization regions on $\alpha_{k-1}(g)$ about their respective output vectors. As an approximation to this error power, we assume that the uniform lattice VQ for $\boldsymbol{u}$ has congruent quantization regions; hence, we need only consider a typical region on the gain quantized cross polytope $\beta_k(\hat{g})$. Let $N_u(\hat{g})$ represent the number of quantization regions for $\boldsymbol{u}$ on $\beta_k(\hat{g})$. Although we actually quantize on $\alpha_{k-1}(\hat{g})$, one face of $\beta_k(\hat{g})$, it is often useful to imagine the total partitioning of the input space into its N regions. Since there are 2^k such faces, we have $2^{-k}N_u(\hat{g})$ regions on the face $\alpha_{k-1}(\hat{g})$. Another way to understand this reduction (by 2^k) is that we must spend k bits to keep the signs of the x_i for the inverse of Helmert's transformation. Let $S_{k-1}(\hat{g})$ represent the typical quantization region for $\boldsymbol{u}$ on $\alpha_{k-1}(\hat{g})$. Ignoring the effects of the edges of $\alpha_{k-1}(\hat{g})$, the second term of (A1) is approximately equal to the number of regions times the inertia of each region,

$$\int_{\alpha_{k-1}(g)} \left[\sum_{j=1}^{k-1} (u_j - \hat{u}_j)^2 \right] d\boldsymbol{u} \approx \frac{N_u(\hat{g})}{2^k} \times \text{inertia}\{S_{k-1}(\hat{g})\}.$$

Substituting these two results,

$$\text{mse} \approx \frac{1}{k} \int_0^{\infty} \left[(g - \hat{g})^2 \times \text{vol}\{\alpha_{k-1}(g)\} + \frac{N_u(\hat{g})}{2^k} \times \text{inertia}\{S_{k-1}(\hat{g})\} \right] 2^{k/2} e^{-\sqrt{2k}g} \, dg.$$

Let U_{k-1} be the inertia of a normalized (unit volume) version of the typical quantization region of the $k-1$ dimensional lattice VQ for $\boldsymbol{u}$ on $\alpha_{k-1}(\hat{g})$. Since we are in dimension $k-1$ we note that scaling the region by a constant a changes the volume by a^{k-1} and the mse by a^{k+1}. Together these yield

$$\text{inertia}\{S_{k-1}(\hat{g})\} = U_{k-1}[\text{vol}\{S_{k-1}(\hat{g})\}]^{(k+1)/(k-1)}.$$

This result and the volume expression

$$\text{vol}\{\alpha_{k-1}(g)\} = \frac{k^{k/2}}{(k-1)!} g^{k-1}$$

result in

$$\text{mse} = \frac{1}{k} \int_0^{\infty} \left[(g - \hat{g})^2 \frac{k^{k/2} g^{k-1}}{(k-1)!} + \frac{N_u(\hat{g}) U_{k-1}}{2^k} \cdot [\text{vol}\{S_{k-1}(\hat{g})\}]^{(k+1)/(k-1)} \right] 2^{k/2} e^{-\sqrt{2k}g} \, d\boldsymbol{u} \, dg.$$

A. Asymptotics

To simplify notation for the asymptotic analysis (as $N \to \infty$), we will replace the sequence $N_u(\hat{g})$ by a levels allocation function $N_u(g)$ and the regions $S_{k-1}(\hat{g})$ by $S_{k-1}(g)$. Further, we assume that the volume of $\alpha_{k-1}(g)$ is divided evenly amongst the $2^{-k}N_u(g)$ regions

$$\text{vol}\{S_{k-1}(g)\} = \frac{\text{vol}\{\alpha_{k-1}(g)\}}{N_u(g)/2^k} = \frac{2^k k^{k/2} g^{k-1}}{N_u(g)(k-1)!}.$$

This yields

$$\text{mse} \approx \int_0^{\infty} \left[(g - \hat{g})^2 + \left(\frac{2^k}{k!} \right)^{2/(k-1)} \frac{U_{k-1} g^2 k^{1+3/(k-1)}}{N_u(g)^{2/(k-1)}} \right] \cdot \frac{(2k)^{k/2}}{k!} g^{k-1} e^{-\sqrt{2k}g} \, dg. \qquad \text{(A.2)}$$

To continue the asymptotic analysis we invoke a standard asymptotic technique: replace the nonuniform scalar quantizer for g by a compandor system [3]. We will assume a compressor function h, expandor h^{-1}, and N_g levels in the scalar uniform quantizer. The first term in the error integral (A.2) is replaced by a standard compandor approximation

$$\int_0^{\infty} (g - \hat{g})^2 f(g) \, dg \approx \frac{1}{12 N_g^2} \int_0^{\infty} \frac{f(g)}{|h'(g)|^2} \, dg,$$

where $f(g)$ is a density function. The result is

$$\text{mse} \approx \frac{(2k)^{k/2}}{12k!N_g^2}\int_0^\infty \frac{g^{k-1}e^{-\sqrt{2k}g}}{|h'(g)|^2}\,dg + \frac{(k-1)(2k)^{k/2}U_{k-1}}{k!}\left(\frac{2^k k^{k/2}}{(k-1)!}\right)^{2/(k-1)} \cdot \int_0^\infty g^{k+1}N_u^{-2/(k-1)}e^{-\sqrt{2k}g}\,dg. \quad \text{(A.3)}$$

We can interpret this mse expression as *Bennett's integral* [3] for the vector quantizer which can be evaluated for a particular set of compressor function, $h(g)$, levels allocation function, $N_u(g)$, and g resolution, N_g.

B. Pyramid Vector Quantizers

For the pyramid vector quantizer, $N_u(g)$ is a constant

$$N_u(g) = \frac{N}{N_g}.$$

For fixed values of k and N, the mse is a function of and can be optimized over the choice of the gain compressor, $h(g)$, and the gain resolution, N_g,

$$\text{mse} \approx \frac{(2k)^{k/2}}{12k!N_g^2}\int_0^\infty \frac{g^{k-1}e^{-\sqrt{2k}g}}{|h'(g)|^2}\,dg + \frac{(k-1)U_{k-1}(2k)^{k/2}}{k!}\left(\frac{2^k k^{k/2}}{(k-1)!}\right)^{2/(k-1)} \cdot \int_0^\infty g^{k+1}N^{-2/(k-1)}N_g^{2/(k-1)}e^{-\sqrt{2k}g}\,dg.$$

Since h appears in only one term, the obvious choice for the compressor is Smith's result [28]

$$h(g) = \frac{\int_0^g x^{(k-1)/3}e^{-\sqrt{2k}x/3}\,dx}{\left(\frac{3}{\sqrt{2k}}\right)^{(k+2)/3}\Gamma\left(\frac{k+2}{3}\right)}.$$

We note that in introducing the pyramid VQ's, Fischer [13] employed a Gaussian (Max–Lloyd) quantizer for g. Here we will continue using the best compressor as before and expect that our performance will be slightly better than Fischer's. Also, our result is for any lattice VQ; as discussed in the text, Fischer employed the A_{k-1} lattice only. Optimizing over the scalar N_g by $\partial\text{mse}/\partial N_g = 0$ yields

$$N_g = \left[\frac{\Gamma\left(\frac{k+2}{3}\right)3^{k+2}}{\Gamma(k+2)12U_{k-1}}\right]^{(k-1)/2k} \frac{[(k-1)!]^{1/k}}{2\sqrt{k}}N^{1/k}$$

and

$$\text{mse}_{\text{Pyramid VQ}} \approx 6[(k+1)kU_{k-1}]^{(k-1)/k}3^{2/k} \cdot \left(\frac{\Gamma\left(\frac{k+2}{3}\right)}{(k-1)!}\right)^{3/k}\left(\frac{1}{12}\right)^{1/k}N^{-2/k}.$$

C. The Unrestricted VQ

In our quantization problem, the total number of representation points in $\mathfrak{R}^k$ is a fixed constant N. As a function of the choices of N_g and $N_u(\hat{g}_i)$, this can be written as

$$\sum_{i=1}^{N_g} N_u(\hat{g}_i) = N. \quad \text{(A.4)}$$

We wish to approximate this constraint equation for large N. Let a typical interval in the scalar quantizer for g have endpoints g_i, g_{i+1} and representation level $\hat{g}_i$. For large N_g the width of this interval, $\Delta_{g,i}$, can be related to the slope of the compressor function $h(g)$

$$g_{i+1} - g_i = \Delta_{g,i} \approx \frac{1}{N_g h'(\hat{g}_i)} \text{ or } \frac{1}{N_g} \approx h'(\hat{g}_i)\Delta_{g,i}. \quad \text{(A.5)}$$

Dividing both sides of (A.4) by N_g and employing the approximation of (A.5) yields the constraint

$$\frac{N}{N_g} \approx \sum_{i=1}^{N_g} N_u(\hat{g}_i)h'(\hat{g}_i)\Delta_{g,i} \approx \int_0^\infty N_u(g)h'(g)\,dg. \quad \text{(A.6)}$$

With this integral constraint we can minimize the VQ's mse over the choice of the g quantizer's resolution, N_g, and compressor, $h(g)$, and the levels allocation function, $N_u(g)$. The following sequence of steps yields optimum choices for these functions:

- append the constraint in (A.6) with a Lagrange multiplier to the mse expression of (A.3) and use Euler's equation (calculus of variations) to choose $N_u(g)$. Scale the result to satisfy (A.6);
- substitute $N_u(g)$ into the mse expression and optimize over the scalar parameter N_g by $\partial\text{mse}/\partial N_g = 0$;
- substitute N_g into the mse and use Hölder's inequality to choose $h(g)$ for minimum mse (hint: use $p = k+2$).

After much manipulation, the optimum choices are

$$N_g = \frac{\sqrt{k}}{2}\left(\frac{1}{12U_{k-1}}\right)^{(k-1)/2k} N^{1/k}, \quad \text{(A.7)}$$

$$N_u(g) = \frac{\sqrt{2}(2k)^{k/2}}{(k-1)!}\left(\frac{k}{k+2}\right)^{k-1}(12U_{k-1})^{(k-1)/2k} \cdot N^{(k-1)/k}g^{k-1}e^{-\sqrt{2k}[(k-1)/(k+2)]g}, \quad \text{(A.8)}$$

and

$$h(g) = 1 - e^{-\sqrt{2k}/[(k+2)]g}. \quad \text{(A.9)}$$

With these choices the mse is

$$\text{mse}_{\text{UVQ}} \approx 2\left(\frac{k+2}{k}\right)^{k+2} U_{k-1}^{(k-1)/k}\left(\frac{1}{12}\right)^{1/k} N^{-2/k}.$$

D. Finite N — For any Lattice

To develop an iterative Lloyd-type algorithm we return to an expression for the mse from before the asymptotic analysis:

$$\text{mse} \approx \frac{1}{k}\int_0^\infty \Big[(g-\hat{g})^2 \times \text{vol}\{\alpha_{k-1}(g)\} + \frac{N_u(\hat{g})}{2^k} \times \text{inertia}\{S_{k-1}(\hat{g})\}\Big] 2^{k/2} e^{-\sqrt{2k}g}\, dg.$$

The expression for the volume of $\alpha_{k-1}(g)$ is the same as before; however, rather than dividing $\alpha_{k-1}(g)$ evenly amongst the $2^{-k}N_u(g)$ regions to find the inertia of $S_{k-1}(g)$, we consider its inertia more precisely. For any lattice structure, since the size of $S_{k-1}(g)$ depends upon the size of $\alpha_{k-1}(g)$, we can write

$$\text{inertia}\{S_{k-1}(\hat{g})\} = \gamma_{k-1}\left(\sqrt{2k}\,\hat{g}\right)^{k+1},$$

with γ_{k-1} being a constant dependent upon the lattice type. Simplifying the mse expression yields

$$\text{mse} \approx \int_0^\infty \left[(g-\hat{g})^2 g^{k-1} + \delta \hat{g}^{k+1}\right] \frac{(2k)^{k/2}}{k!} e^{-\sqrt{2k}g}\, dg,$$

where

$$\delta = \frac{N_u(\hat{g})k!\sqrt{2}\,\gamma_{k-1}}{2^{k/2}\sqrt{k}}.$$

To continue we extract the term g^{k-1} from both terms in brackets to make the outer term look like the marginal density of g. Note that to do this we approximate $\hat{g}$ by g in the second term

$$\hat{g}^{k+1} \to \hat{g}^2 g^{k-1}.$$

Separating into individual intervals on g yields

$$\text{mse} \approx \sum_{i=1}^{N_g} \int_{g_i}^{g_{i+1}} \left[(g-\hat{g}_i)^2 + \delta_i \hat{g}_i^2\right] \cdot \frac{(2k)^{k/2}}{k!} g^{k-1} e^{-\sqrt{2k}g}\, dg,$$

where we have indexed the δ by i since the factor depends upon the individual u resolutions. This is the expression from which we develop the standard Lloyd method I iteration:

- $\partial/\partial \hat{g}_i = 0$ yields

$$\hat{g}_i = \frac{\int_{g_i}^{g_{i+1}} g^k e^{-\sqrt{2k}g}\, dg}{(1+\delta_i)\int_{g_i}^{g_{i+1}} g^{k-1} e^{-\sqrt{2k}g}\, dg};$$

- $\partial/\partial g_i = 0$ yields

$$g_i = \frac{\hat{g}_i^2(1+\delta_i) - \hat{g}_{i-1}^2(1+\delta_{i-1})}{2(\hat{g}_i - \hat{g}_{i-1})}.$$

Similarly we can consider the case of a single small region centered at the origin ($\hat{g}_1 = 0$). In this case equating the partial derivative with respect to g_2 to zero yields

$$g_2 = \frac{k+1}{k-1}\hat{g}_2\left(-1+\sqrt{\frac{k(2+\delta_2)-\delta_2}{k+1}}\right).$$

References

[1] J. P. Adoul, "La quantification vectorielle des signaux: Approche algébrique," *Ann. Télécommun.*, vol. 41, no. 3–4, pp. 158–177, 1986.

[2] C. Bei and R. M. Gray, "An improvement of the minimum distortion encoding algorithm for vector quantization," *IEEE Trans. Commun.*, vol. COM-33, pp. 1132–1133, Oct. 1985.

[3] W. R. Bennett, "Spectra of quantized signals," *Bell System Tech. J.*, vol. 27, pp. 446–472, Mar. 1948.

[4] J. A. Bucklew and N. C. Gallagher, "Quantization schemes for bivariate Gaussian random variables," *IEEE Trans. Inform. Theory*, vol. IT-25, no. 5, pp. 537–543, Sept. 1979.

[5] ——, "Two-dimensional quantization of bivariate circularly symmetric densities," *IEEE Trans. Inform. Theory*, vol. IT-25, no. 6, pp. 667–671, Nov. 1979.

[6] J. A. Bucklew, "Companding and random quantization in several dimensions," *IEEE Trans. Inform. Theory*, vol. IT-27, no. 2, pp. 207–211, Mar. 1981.

[7] A. Buzo, A. H. Gray, R. M. Gray, and J. D. Markel, "Speech coding based upon vector quantization," *IEEE Trans. Acoust. Speech Signal Processing*, vol. ASSP-28, pp. 562–574, Oct. 1980.

[8] D. Y. Cheng, A. Gersho, B. Ramamurthi, and Y. Shoham, "Fast search algorithms for vector quantization and pattern matching," in *Proc. IEEE ICASSP*, 1985, pp. 9.11.1–9.11.4.

[9] J. H. Conway and N. J. A. Sloane, "Voronoi regions of lattices, second moments of polytopes, and quantization," *IEEE Trans. Inform. Theory*, vol. IT-28, no. 2, pp. 211–226, Mar. 1982.

[10] ——, "Fast quantizing and decoding algorithms for lattice quantizers and codes," *IEEE Trans. Inform. Theory*, vol. IT-28, no. 2, pp. 227–232, Mar. 1982.

[11] H. S. M. Coxeter, *Regular Polytopes*. New York: Dover, 1973.

[12] T. R. Fischer and R. M. Dicharry, "Vector quantizer design for Gaussian, gamma, and Laplacian sources," *IEEE Trans. Commun.*, vol. COM-32, pp. 1065–1069, Sept. 1984.

[13] T. R. Fischer, "A pyramid vector quantizer," *IEEE Trans. Inform. Theory*, vol. IT-32, no. 4, pp. 568–583, July 1986.

[14] A. Gersho, "Asymptotically optimal block quantization," *IEEE Trans. Inform. Theory*, vol. IT-25, no. 4, pp. 373–380, July 1979.

[15] J. D. Gibson and K. Sayood, "Lattice quantization," *Advances in Electronics and Electron Phys.*, vol. 72, pp. 259–330, 1988.

[16] B. H. Juang and A. H. Gray, "Multiple stage vector quantization for speech coding," in *Proc. IEEE ICASSP*, 1982, pp. 597–600.

[17] M. G. Kendall, *The Geometry of n Dimensions*. London: Charles Griffin and Co., 1961.

[18] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. COM-28, pp. 84–95, Jan. 1980.

[19] S. P. Lloyd, "Least squares quantization in PCM," *Trans. IEEE Inform. Theory*, vol. IT-28, no. 2, pp. 129–137, Mar. 1982.

[20] T. D. Lookabaugh and R. M. Gray, "High-resolution quantization theory and the vector quantizer advantage," *Trans. IEEE Inform. Theory*, vol. 35, no. 5, pp. 1020–1033, Sept. 1989.
[21] P. Noll and R. Zelinski, "Comments on 'Quantizing characteristics for signals having Laplacian amplitude probability density function'," *IEEE Trans. Commun.*, vol. COM-27, pp. 1259–1260, Aug. 1979.
[22] ____, "Bounds on quantizer performance in the low bit-rate region," *IEEE Trans. Commun.*, vol. COM-26, pp. 300–304, Feb. 1978.
[23] M. D. Paez and T. H. Glisson, "Minimum mean-square-error quantization in speech PCM and DPCM systems," *IEEE Trans. Commun.*, vol. COM-20, pp. 225–230, April 1972.
[24] W. A. Pearlman, "Polar quantization of a complex Gaussian random variable," *IEEE Trans. Commun.*, vol. COM-27, pp. 892–899, June 1979.
[25] R. C. Reininger and J. D. Gibson, "Distributions of the two-dimensional DCT coefficients for images," *IEEE Trans. Commun.*, vol. COM-31, pp. 835–839, June 1983.
[26] M. J. Sabin and R. M. Gray, "Product code vector quantizers for speech waveform coding," in *Conf. Rec. Globecom*, 1982, pp. E.6.5.1–5.
[27] K. Sayood, J. D. Gibson, and M. C. Rost, "An algorithm for uniform vector quantizer design," *IEEE Trans. Inform. Theory*, vol. IT-30, no. 6, pp. 805–814, Nov. 1984.
[28] B. Smith, "Instantaneous companding of quantized signals," *Bell Sys. Tech. J.*, vol. 36, pp. 653–709, Mar. 1957.
[29] P. F. Swaszek and J. B. Thomas, "Multidimensional spherical coordinates quantization," *IEEE Trans. Inform. Theory*, vol. IT-29, no. 4, pp. 570–576, July 1983.
[30] P. F. Swaszek and T. W. Ku, "Asymptotic performance of unrestricted polar quantizers," *IEEE Trans. Inform. Theory*, vol. IT-32, no. 2, pp. 330–333, Mar. 1986
[31] P. F. Swaszek, "Vector quantization for image compression," in *Proc. Princeton CISS*, Mar. 1986, pp. 254–259.
[32] S. G. Wilson, "Magnitude/phase quantization of independent Gaussian variates," *IEEE Trans. Commun.*, vol. COM-28, pp. 1924–1929, Nov. 1980.
[33] P. L. Zador, "Development and evaluation of procedures for quantizing multivariate distributions," Ph.D. dissert., Stanford Univ., Dept. of Statis., Stanford, CA, 1963.
[34] Y. Zhu, *Vector Quantization for Data Compression*, MSEE thesis, Dept. Elect. Eng., Univ. Rhode Island, Kingston, 1989.