A METHOD FOR EXAMINING VECTOR QUANTIZER STRUCTURES

Erik Agrell
Department of Information Theory
Chalmers University of Technology
S–412 96 Göteborg, Sweden

Abstract — **This paper presents how to study the geometry of Voronoi regions in an arbitrary vector quantizer. Methods to find the location, the extent, and the faces of any region are summarized.**

I. INTRODUCTION

It is well known [4] that a vector quantizer (VQ) performs better, in terms of signal-to-noise ratio, than a scalar quantizer. The improvement increases with the dimension, but the price paid is complexity. In particular, the encoding process is slower. In the case of *nearest neighbor* quantization, which this presentation considers, the straightforward encoding method is calculating the squared Euclidean distance

$$d(\mathbf{w}, \mathbf{r}_i) = \|\mathbf{w} - \mathbf{r}_i\|^2 \tag{1}$$

between an input vector $\mathbf{w}$ and every reconstruction vector $\mathbf{r}_i; i = 1, \ldots, n$, and selecting the codeword that gives the minimum distance. The set of vectors that are encoded as a certain codeword j according to this rule is called the *Voronoi region* (VR)

$$V_j = \left\{ \mathbf{w} : d(\mathbf{w}, \mathbf{r}_j) \le d(\mathbf{w}, \mathbf{r}_i); \quad i = 1, \ldots, n \right\} \tag{2}$$

Sometimes suboptimal VQs are accepted in order to decrease the encoding time. Several structures have been developed for which fast search algorithms exist, e.g. lattice or multi-stage coders. A cruder case is scalar quantization of every component, which is very convenient in this aspect, although the advantages of vector quantization are surrendered.

However, there are also methods to improve the encoding speed for arbitrary vector quantizers, without paying with signal-to-noise ratio. Such methods often require precomputing some geometrical properties of the VRs. A new method to obtain such information is presented here, as well as an encoding algorithm based on the precomputed data.

II. EXAMINING THE GEOMETRY OF VORONOI REGIONS

Some relevant types of problems concerning the structure of given VRs are:

1. What values of a certain component may vectors in this VR take on?
2. On which side of a certain hyperplane lies this VR, or is the VR intersected?
3. Have these two VRs a common face?

The three questions are closely related. All of them have applications in different algorithms for the design of fast encoders, see below. Probabilistic methods have been proposed to obtain approximate, or likely, answers to them [2], [3]. In this section, deterministic methods, based on different applications of linear programming, are presented for solving these and related problems reliably.

Consider the following standard formulation of a linear programming problem:

$$\min \mathbf{c}^T \mathbf{x} \quad \text{when} \begin{cases} A\mathbf{x} = \mathbf{b} \\ \mathbf{x} \ge \mathbf{0} \end{cases} \tag{3}$$

Much research and much literature have been devoted to methods for solving it. Two of the main approaches are the Simplex Method and Karmarkar's algorithm, both having numerous variations [1]. From optimization theory it is known that there exists a *dual* problem to (3),

$$\max \mathbf{b}^T \mathbf{w} \quad \text{when } A^T \mathbf{w} \le \mathbf{c} \tag{4}$$

the solution of which is closely related to the solution of (3). Actually, both mentioned methods generate the solution of (4) as a by-product when solving (3).

The inequality constraints in (4) form a convex polytope. They describe the VR V_j (2) of a certain codeword j if

$$A = \left[\mathbf{a}_1 \ldots \mathbf{a}_{j-1}\ \mathbf{a}_{j+1} \ldots \mathbf{a}_n \right] \qquad \mathbf{c} = \left[c_1 \ldots c_{j-1}\ c_{j+1} \ldots c_n \right] \tag{5a}$$

where for every i

$$\mathbf{a}_i = \mathbf{r}_i - \mathbf{r}_j \qquad c_i = \frac{\|\mathbf{r}_i\|^2 - \|\mathbf{r}_j\|^2}{2} \tag{5b}$$

Thus, the dual problem can be used for finding extrema of a VR.

Depending on the choice of the dual objective vector $\mathbf{b}$, different extrema will be found and different properties of the VR will be investigated. Especially, if $\mathbf{b}$ is chosen first as a unit vector and next as the same unit vector negated, problem 1 above is solved by two linear programs. If this is repeated for all coordinates and all VRs, a circumscribed hyperrectangle will be found for each VR, which is the precomputed information required for encoding with the *Projection Method* [2].

Problem 2 is solved similarly by two linear programs, if $\mathbf{b}$ is set orthogonal to the given hyperplane, pointing in both directions. If these two extrema lie on the same side of the given hyperplane, so does the whole VR; otherwise it is intersected. The answer to this kind of questions is vital for the design of the decision tree used in the *Binary Hyperplane Testing Algorithm* [3].

Finally, to test the neighborship between VRs j and k (problem 3), $\mathbf{b}$ is chosen equal to $\mathbf{a}_k$, which is orthogonal to the common face of V_j and V_k, if such a face exists, whereas A and $\mathbf{c}$ as before denote V_j (5). With this input, a linear programming algorithm will return the point $\hat{\mathbf{w}}$ in V_j whose projection on $\mathbf{a}_k$ is closest to $\mathbf{r}_k$. If the two VRs have a common face, the dual optimum $\hat{\mathbf{w}}$ will inevitably lie on it. The primal optimum shows whether this has occurred: the face was reached if and only if the component of $\hat{\mathbf{x}}$ corresponding to $\mathbf{a}_k$ is greater than zero.

A table of all neighborships in a VQ is a useful tool both for theoretical analysis and in applications. One application is the fast encoding algorithm described below.

III. THE "NEIGHBOR DESCENT" ENCODING METHOD

A VR is defined by $n-1$ linear inequalities as in (2) or (4). Some of them are in general redundant. Define the set N_j of neighbors to a codeword j as all codewords whose VRs have a face in common with V_j. The corresponding inequalities are the only ones needed to be considered in order to determine if a vector $\mathbf{w}$ belongs to a certain VR V_j:

$$V_j = \left\{ \mathbf{w} : \mathbf{a}_i^T \mathbf{w} \le \mathbf{c}_i; \quad i \in N_j \right\} \tag{6}$$

This fact can be used to speed up encoding. Since the method described in the previous section now makes it possible to compute a complete neighborship table N_j for every codeword in a moderate-size VQ, an encoding algorithm that makes use of these tables is of more than theoretical interest. One such method, called *neighbor descent*, is introduced here. It can be used independently or to improve a solution given by any suboptimal encoding algorithm.

Suppose that a vector $\mathbf{w}$ is to be encoded and that there is reason to believe that $\mathbf{r}_j$ is a good reconstruction vector for $\mathbf{w}$. (If no such knowledge is available, choose instead a codeword j whose reconstruction vector is fairly central in relation to the others.) Calculate the distance $d(\mathbf{w}, \mathbf{r}_i)$ between $\mathbf{w}$ and $\mathbf{r}_i; i \in N_j$, the neighbors of j, one at a time. As soon as a neighbor i is encountered that has a smaller distortion (distance to $\mathbf{w}$) than j, replace j with i and restart. If no better codeword is found in N_j, then stop.

Theorem of uniqueness: In any VQ, for any input $\mathbf{w}$, no more than one codeword can have a smaller distortion than all its neighbors.

A necessity for the success of the method is that a path through neighboring VRs, along which the distortion $d(\mathbf{w}, \mathbf{r}_j)$ is monotonically decreasing, does not terminate in a suboptimal local minimum. The above theorem states that this can never be the case. Its proof follows as a direct consequence of (6) and the observation that a vector cannot belong to the interior of more than one VR.

The performance of the neighbor descent method was evaluated in experiments on vector quantizers without an induced structure. A table of computed distances was maintained during the search to avoid duplication of work. The results show that many of the n distance calculations can be avoided when the neighbor descent method is used. The reduction is greatest for VQs with high bit rates. The speed of the method can be further increased by combining it with known ways to speed up distance calculations, such as the *Partial Distance Method* [2], [4].

REFERENCES

[1] M. S. Bazaraa, J. J. Jarvis, and H. D. Sherali, *Linear Programming and Network Flows*. New York: Wiley, 1990.
[2] D.-Y. Cheng, A. Gersho, B. Ramamurthi, and Y. Shoham, "Fast Search Algorithms for Vector Quantization and Pattern Matching," in *Proc. Int. Conf. Acoust., Speech, Signal Processing*, pp. 9.11.1–4, San Diego 1984.
[3] D.-Y. Cheng and A. Gersho, "A Fast Codebook Search Algorithm for Nearest-Neighbor Pattern Matching," in *Proc. Int. Conf. Acoust., Speech, Signal Processing*, pp. 265–268, Tokyo 1986.
[4] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Boston: Kluwer, 1992.